Entropy-constrained Reflected Residual Vector Quantization with Application to Image Coding

Wail Abdul-Hakim Mousa

Electrical Engineering

April 2003

Dedicated to

My beloved Father and Mother

and

My Wife

ACKNOWLEDGEMENTS

In the name of Allah, the Most Gracious and the Most Merciful

All praise and glory goes to Almighty Allah (Subhanahu Wa Ta'ala) who gave me the courage and patience to carry out this work. Peace and blessings of Allah be upon His last Prophet Muhammad (Sallulaho-Alaihi-Wassalam) and all his Sahaba (Radhi-Allaho-Anhum) who devoted their lives towards the prosperity and spread of Islam.

Acknowledgement is due to King Fahd University of Petroleum and Minerals represented in the department of Electrical Engineering for providing support for this work.

My deep appreciation and gratitude goes to my old thesis advisor Dr. Mohammed A. U. Khan for his constant endeavor, guidance and the numerous moments of attention he devoted throughout the course of this research work. His valuable guidance made it possible for me to finish this work.

I extend my deepest gratitude to my current thesis advisor Dr. Omar A. Al-Swailem who has taken over after Dr. Khan left KFUPM. In fact, Dr. Swailem was the first person who introduced me to the digital image processing field and generated my interest in doing research in this area.

Also, I would like to pass many thanks to my thesis committee members Dr. Samir H. Abdul-Jauwad, and Dr. Ahmed A. Massoud, for their encouragement and cooperation.

Special thanks are due to Dr. Sami Khyyat, Dr. Rafat Nassar, Dr. Waleed Al-Sabah, Dr. Hussain Al-Duwaish, Dr. Zakriya Al-Homouz, Dr. Thaheem, Dr. Y. Figabedzi, Mr. L. Ghouti, Dr. Samir. Al-Byiat, Dr. A. Zergine, Dr. Mohammad. Al-Sunaidi, Dr. Hussian Messoudi, Dr. Adil S. Balghoname, Dr. Talal Halwani, Dr. M. H. Al-Shwaihdi, Dr. Saleem Messoudi, and all the people who supported me during my studies at KFUPM and made my work and stay at KFUPM very pleasant and joyful.

I cannot forget the prayers and full support of my father Professor Abdul-Hakim Mousa and my Mother, in addition to my sister's and brothers Abdullah and Mohammed. Also, I cannot forget the prayers of my uncles: Abdullah, Ahemd, Khaled, and my grandmother Alaweyah. In addition to that, I won't forget my father and mother in laws for their prayers. To my grandfather Mohammad Khttab Shaker, Mousa, and may grandmother A'mena, may their souls rest in peace and may they earn a high place in paradise. (Amen)

Contents

Acknowledgements **ii**

List of Tables **vii**

List of Figures **xi**

Abstract (English) **xii**

Abstract (Arabic) **xiii**

1 Introduction **1**

1.1 Introduction 1

1.2 Scope of the Thesis 6

2 Background **9**

2.1 Introduction 9

2.2 Rate-Distortion Theory 10

2.3 Vector Quantization (VQ) 12

2.3.1 Introduction 12
2.3.2 Description of VQ 13
2.3.3 VQ Design Techniques 18
2.3.4 Complexity of VQ 20
2.3.5 Performance of VQ 21
2.4 Constrained VQ 21
2.4.1 Tree-Structured VQ 23
2.4.2 Lattice VQ 24
2.4.3 Product Code VQ 25
2.4.4 Residual VQ 26
2.5 Distortion Measures 27
2.6 Image Sources 31

3 Fixed Rate Reflected Residual Vector Quantization 33

3.1 Some Preliminaries 33
3.1.1 A Review of Residual Vector Quantizer Structure 33
3.1.2 Direct Sum Quantizers 34
3.1.3 Entanglements 37
3.2 Reflected RVQ (RRVQ) 39
3.3 The Fixed Rate Design Algorithm 45
3.4 Performance of Fixed Rate RRVQ 46

4 Entropy-constrained Reflected Residual Vector Quantization 50

4.1 Introduction 50

4.2 The EC-RRVQ structure 52

4.2.1 Operation of the EC-RRVQ Algorithm 55

4.2.2 Complexity of the EC-RRVQ Algorithm 57

4.3 EC-RRVQ Performance 61

4.3.1 Synthetic Sources 61

4.3.2 Natural Images 62

4.3.3 Extension to large block EC-RRVQ 76

5 Conclusion 88

5.1 Summary of Thesis Contributions 88

5.2 Future Research Directions 89

APPENDICES 91

A Mid-point derivation in the Lagrangian space 91

Nomenclature 94

List of Tables

4.1 PSNR OF EC-RVQ AND EC-RRVQ FOR FOUR TEST IMAGES TAKEN FROM THE USC DATABASE (THE BIT RATE IS 0.5 BPP AND THE VECTOR SIZE IS 4×4) 72

List of Figures

1.1 Basic digital communication system 3

2.1 Vector Quantizer Procedure 15

3.1 A two-stage Residual Vector Quantizer 35

3.2 Three-stage scalar binary RVQ tree structure 38

3.3 Three-stage scalar RRVQ tree structure 40

3.4 Gaussian source coded with a binary 8-stage, two-dimensional quantizer. (a) Equivalent code vector constellation of RVQ. (b) Code vector constellation of RRVQ. 43

3.5 Performance comparison of RRVQ with RVQ for the memoryless Gaussian source 48

3.6 Performance comparison of RRVQ with RVQ for the memoryless Laplacian source 49

4.1 The EC-RRVQ design algorithm 58

4.2 Encoding complexity of EC-RVQ and EC-RRVQ 60

4.3 Performance comparison of EC-RRVQ with various source coding schemes for the memoryless Gaussian source 63

4.4 Performance comparison of EC-RRVQ with various source coding schemes for the memoryless Laplacian source 64

4.5 PSNR performance for the test image LENA using both jointly and non-jointly optimized decoders (The number of stages is 16 and $m = 1$) 66

4.6 Rate-distortion performance of EC-RRVQ with 16 stages for the test image LENA at increasing values of m (The vector size is 4×4) . . . 68

4.7 Rate-distortion performance of EC-RRVQ for the test image LENA at two different peak bit rates. The top one is for 28 stages giving 1.75 bpp and the bottom curve is for 16 stages giving 1.00 bpp (The vector size is 4×4 and $m = 1$) 69

4.8 Rate-distortion performance of EC-RRVQ showing the rates at which stages were unreflected for the test image LENA . The number of stages was 16 at an increasing values of m (The vector size is 4×4) . 71

4.9 Rate-distortion performance of EC-RRVQ and EC-RVQ with 16 stages for the test image LENA at $m = 1$ (The vector size is 4×4) 73

4.10 Cropped Image LENA coded using (b) EC-RVQ at a bit rate of 0.527 bpp with PSNR of 30.61 dB (c) EC-RRVQ at a bit rate of 0.526 bpp with PSNR of 31.41 dB 74

4.11 Image BOAT coded using (b) EC-RVQ at a bit rate of 0.641 bpp with PSNR of 28.82 dB (c) EC-RRVQ at a bit rate of 0.593 bpp with PSNR of 29.2 dB 75

4.12 Rate-distortion performance of EC-RRVQ with 32 stages for the test image LENA at increasing values of m (The vector size is 8×8) . . . 80

4.13 Cropped Image LENA coded using EC-RRVQ of dim=8×8 and $m = 1$ at a bit rate of (a) 0.349 bpp with PSNR of 30.49 dB (b) 0.257 bpp with PSNR of 28.99 dB (c) 0.177 bpp with PSNR of 28.15 dB (d) 0.129 bpp with PSNR of 26.29 dB 81

4.14 Rate-distortion performance of EC-RRVQ and EC-RVQ with 32 stages for the test image LENA at $m = 1$ (The vector size is 8×8) 82

4.15 Cropped Image LENA coded using (a) EC-RVQ at a bit rate of 0.179 bpp with PSNR of 28.03 dB (b) EC-RRVQ at a bit rate of 0.177 bpp with PSNR of 28.15 dB both of dimension 8×8 and $m = 1$ 83

4.16 Rate-distortion performance of EC-RRVQ with 64 stages for the test image LENA at increasing values of m (The vector size is 16×16) . . 84

4.17 Cropped Image LENA coded using EC-RRVQ of dim=16×16 and $m = 1$ at a bit rate of (a) 0.201 bpp with PSNR of 29 dB (b) 0.164 bpp with PSNR of 27.83 dB (c) 0.106 bpp with PSNR of 26.11 dB (d) 0.066 bpp with PSNR of 24.76 dB 85

4.18 Rate-distortion performance of EC-RRVQ and EC-RVQ with 64 stages for the test image LENA at $m = 1$ (The vector size is 16×16) 86

4.19 Cropped Image LENA coded using (a) EC-RVQ at a bit rate of 0.215 bpp with PSNR of 28.39 dB (b) EC-RRVQ at a bit rate of 0.201 bpp with PSNR of 29 dB both of dimension 16×16 and $m = 1$ 87

THESIS ABSTRACT

Name: WAIL ABDUL-HAKIM MOUSA

Title: ENTORPY-CONSTRAINED REFLECTED VECTOR QUANTIZATION WITH APPLICATION TO IMAGE CODING

Degree: MASTER OF SCIENCE

Major Field: ELECTRICAL ENGINEERING

Date of Degree: April 2003

An entropy-constrained reflected residual vector quantization (EC-RRVQ) design algorithm is introduced as an alternative to entropy-constrained residual vector quantization (EC-RVQ) and used to design codebooks for image coding. EC-RRVQ is able to realize a more ordered or less random codebook, that offers two advantages. The first is that an ordered codebook have lowe output entropy while the second has to do with simplifying the search procedure used to find the best codeword. The idea discussed in this work is to introduce EC-RRVQ as a new baseline quantization scheme with single-path search and improved rate-distortion performance, which if joined with other transform and subband coding methods will result in a competitive design. Because of single-path search, the EC-RRVQ has the potential to be a serious contender in the list of large block vector quantization implementation algorithms. Experimental results indicate that good image reproduction quality can be accomplished at relatively low bit rates. The performance of 16×16 *EC-RRVQ at* 0.2 *bpp is* 29 *dB as compared to* 28.39 *dB for EC-RVQ with same dimensions and bit rate.*

Keywords: *Vector Quantization (VQ), RVQ, RRVQ, EC-RRVQ, EC-RVQ.*

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS, DHAHRAN.
APRIL 2003

Chapter 1

Introduction

1.1 Introduction

Systems which are dedicated to the communication or storage of information are commonplace in everyday life. Examples of typical signals of interest are speech and audio, digital images, high quality medical images, and digital video. Nowadays, advances in communication and memory technologies have led to communication channels with relatively large capacities and computers with extremely large memories. However, the availability of good compression techniques is recognized as being curical to a wide range of developing applications. Compression is concerned with making efficient use of storage media and communication resources.

A common objective of data compression systems is to reduce the number of bits used to represent the data while one can still be able to convey the perti-

nent information. Rate distortion theory [1], which is considered to be a branch of information theory [2], provides an extensive mathematical treatment of the data compression problem. Unfortunately, it does not take into account the complexity and implementation of practical data compression systems. Nonetheless, the theory is very important in the sense that it provides Lowe bounds on the achievable rate-distortion performance for data compression systems.

A basic communication system is shown in Fig. 1.1. At the sending end, a continuous-time signal is first converted to a discrete-time signal based on Shanonn's sampling theorem [2]. Then the output of such block is quantized in order to have a finite amplitude values. By quantizing such signal, an amount of distortion is introduced to the signal as it is an irreversible process. The final block sending end is the channel encoding process. The purpose of the channel encoding block is to add extra bits to provide protection to the information conveyed through the channel. At the receiving end, channel decoding will provide error-detection and/or error-correction to the received data. Quantization here will approximately reconstruct the amplitude values and interpolation will be done to the output of the quantizer to get a continuous-time signal.

Vector Quantization (VQ) is the extension of scalar quantization to higher dimensional space. VQ has received much attention as a powerful speech and image compression technique [3], [4], [5]. It has the ability to exploit the memory or correlation between neighboring pixels in an image. This is based on the fundamental

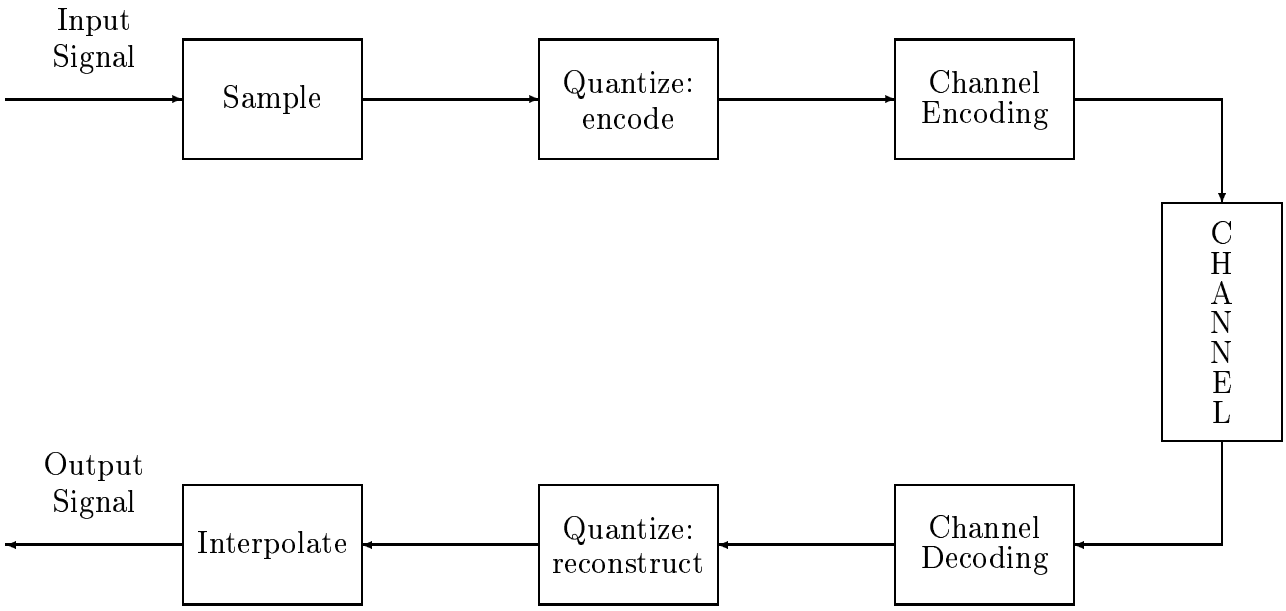


Figure 1.1: Basic digital communication system

result from Shannon information theory which states that better performance can be achieved by coding vectors instead of scalars [6]. As a result, the performance of a vector quantizer can approach the rate-distortion bound $R(D)$ as the vector size increases [4]. However, an issue of recognized importance for VQ is that the size of the codebook grows exponentially as a function of the vector size and the bit rate. Accordingly, both the computation and memory requirements associated with VQ will increase exponentially. Therefore, unconstrained exhaustive search VQ image coding has been limited to a relatively small vector size such as 4×4.

To overcome the complexity and memory barriers, many researchers have suggested imposing certain structural constraints on the VQ codebook design. By doing so, higher vector dimensions and large codebook sizes become feasible. However, the reduction in complexity is often a good trade for only a moderate loss in quality. Examples of such structured VQ's are tree-structured VQ's (TSVQ's) and lattice VQ's [4]. Residual VQ (RVQ) is one of the simple and efficient types of structurally constrained VQ designs [7], [8]. An RVQ consists of multiple stages of VQ where each stage is operating on the *residual* of the previous stage. RVQ is a design that is capable of reducing both memory and computation costs [4]. In addition, it can operate over a large range of bit rates and vector dimensions. A simple two-stage RVQ was proposed by Juang and Gray in 1982 for coding of speech signals [9]. This RVQ encoder utilizes a computationally inexpensive, but suboptimal sequential single-path search. The performance of this computationally cheap RVQ was

found to degrade significantly as the number of stages grows beyond two.

Later on, the RVQ received more interest and advances in both theory and design [10], [7], [8]. In 1989, Barnes and Frost [7], introduced a jointly optimized RVQ or simply RVQ design. In their design, an attempt was made to minimize the overall quantization error of the RVQ instead of merely optimizing the individual stages in isolation. They demonstrated that a sequential single-path search through VQ stages cannot, in general, utilize all the available codevectors. They employed M-search, an efficient multi-path tree search algorithm, to search the stage codebooks. The design resulted in improvement in terms of rate-distortion performance. However, the increase in performance came at the expense of additional computations.

Moreover, many variable rate vector quantization based algorithms have been considered previously for image coding. Examples of variable rate VQ's are pruned tree-structured VQ (PTSVQ), and entropy-pruned tree-structured VQ (EP-TSVQ) [11]. It was shown in [12] that the rate-distortion performance of a RVQ can be further improved by including entropy encoding which is a variable rate VQ. The method was referred to as Entropy-constrained RVQ (EC-RVQ). Experimental results, reported in [12], show that EC-RVQ outperforms single-stage entropy-constrained VQ (ECVQ) [13]. Nevertheless, the RVQ and EC-RVQ both suffers from computational costs. An option for lowering RVQ encoding complexity cost is the imposition of additional structure on the RVQ stage codebooks to make the code more submissive to sequential, single path searches. Multiple-stage VQ's with

stage codebooks comprised of lattice VQ's [14] and reflected RVQ (RRVQ) [10] are examples of this option. Our focus in this thesis is on the RRVQ.

A reflected RVQ (RRVQ) is a multistage structure with binary stage codebooks. The encoder and decoder of RRVQ performs a *reflecting* or *folding* operations on the residual vectors between residual stages. This folding operation forces a certain symmetry on the RVQ codebook. In other words, the structure of the RRVQ direct sum codebook or its constellation will retain no diffusion between the codevectors which in turn makes the sequential search of stage codebooks optimal [10] and [8]. The experimental results, reported in [10], show that the imposition of reflection constraint lead to an unavoidable increase in distortion as compared to RVQ. However, since structured systems are inherently *less random* or *more ordered*, it was conjectured that in general, an RRVQ also have lower output entropy as compared to RVQ. Recently, an entropy-constrained RRVQ (EC-RRVQ) design was introduced based on the conjecture, and experimental results validated the conjecture [15].

1.2 Scope of the Thesis

This thesis deals with the complexity with trade off rate-distortion performance Vector Quantizater. The idea discussed in this work is to introduce entropy-constrained RRVQ (EC-RRVQ) as a new baseline quantization scheme. EC-RRVQ has the property of using single-path search, and hence have a cheap encoder complexity, and

improved rate-distortion performance. The work presented in this thesis has the following major contributions:

- We will show that, by including noiseless output entropy on the RRVQ design, EC-RRVQ will provide a notable improvement over the fixed rate RRVQ, fixed rate RVQ. Also, the EC-RRVQ will outperform the previous entropy-constrained RVQ (EC-RVQ) in terms of rate-distortion performance, computational complexity and memory requirements.

- Since EC-RRVQ uses single-path search, we will illustrate that the EC-RRVQ has the potential to be a serious contender in the list of large block vector quantization implementation algorithms.

- Although monotonic convergence properties are lost, the EC-RRVQ design algorithm implemented in this work will be a stable algorithm as compared to a similar algorithm (a scalar case of EC-RRVQ) reported in [16].

In this thesis, some background is introduced about Rate Distortion Theorem. Unconstrained Vector Quantization is described. Then Constrained VQ is introduced.

Chapter 3 of this thesis deals with the fixed rate RRVQ. Some preliminaries about RVQ is first given. Then, the RRVQ theory is fully investigated.

In Chapter 4, we consider the utility of of the EC-RRVQ reported in [15] as an image coding method applied directly to image pixels. The idea is to introduce EC-RRVQ as a new base-line quantization scheme with reduced encoding complexity,

that if joined with other transform and subband coding methods will result in a competitive design. It basically introduces the EC-RRVQ algorithm and discusses the operation of the algorithm. Finally, the encoder complexity is presented. The EC-RRVQ performance is fully discussed and compared with previous algorithms. Investigation of the extension to large block sizes such as 8×8, and 16×16 is also presented.

Finally, Chapter 5 concludes with a summary that highlights the advantages and disadvantages of the EC-RRVQ method and suggests some possible directions for further studies.

Chapter 2

Background

2.1 Introduction

Working with digital images often involves storing them in short-term memory (RAM) for convenient access, processing them on a pixel-by-pixel basis, and archiving the processed results or transmitting them to another location. Digital image compression represents an immediate and practical approach to help address storage limitations and transmission channel bottlenecks [17]. Data compression has been studied for years by many information theoreticians throughout the world. On a conceptual level, the basic principles are simple. Images can be compressed because they contain redundancies. By removing such redundancies, the number of bits required to represent the image can be reduced. Since VQ and its other forms are considered to be lossy compression methods, it is useful to introduce some theoret-

ical background regarding lossy compression. More specifically, it is necessary to introduce basic elements of *information theory* and *rate-distortion theory*.

2.2 Rate-Distortion Theory

Consider samples that are *discrete-time*, *discrete-valued* random variable $\boldsymbol{X}$ with a finite *N-symbol* alphabet $\{x_1, x_2, \cdots, x_N\}$ with probability $pr(x_i)$. The *information* or *self-information* may be defined formally as $[-\log_2 pr(x_i)]$. The average information is called the *first-order entropy*, denoted $H(x)$. The first-order entropy is defined formally as the expected value of the self-information:

$$H = E[-\log_2 pr(x_i)] = -\sum_{i=1}^{N} pr(x_i) \log_2 pr(x_i) \tag{2.1}$$

Furthermore, assume that the input is *stationary* random sequence, x_n, which means that the statistics of x_n do not change with time. More precisely, by given the probability of the k-dimensional random vector $\boldsymbol{X} = (x_1, x_2, \cdots, x_k)$, the value $\boldsymbol{X} = (x_{j+1}, x_{j+2}, \cdots, x_{j+k})$ is independent of the initial index j. The entropy of $\boldsymbol{X}$ is given by:

$$H(\boldsymbol{X}) = -\sum_{\boldsymbol{x}} pr(\boldsymbol{x}) \log_2 pr(\boldsymbol{x}) \tag{2.2}$$

where $\boldsymbol{x}$ is a realization of $\boldsymbol{X}$. The most important measure of a stationary source is its entropy rate $H_{\boldsymbol{X}}$ defined as

$$H_{\boldsymbol{X}} = \frac{1}{k} \lim_{k \to \infty} H(x_1, x_2, \cdots, x_k) \tag{2.3}$$

It is shown in [2], [18] that if the source is stationary, then the limit always exists and is approached monotonically based on Eq. 2.3 as $k \to \infty$. Generally speaking, the entropy rate is virtually impossible to determine for natural signals. However, first and higher order entropies can be computed and used in practical signal compression systems. If the source symbols were, in addition to be stationary, *independent and identically distributed* (i.i.d) random variables (discrete memoryless source), then the first-order entropy is equivelant to the entropy rate [31].

Rate-distortion theory is the branch of information theory which addresses situations where lossy compression is necessary. In other words, where the entropy of the source exceeds the channel capacity. It provides a useful mathematical formalism in which compression may be analyzed and a useful criterion upon which compression schemes may be based. The Rate-distortion function $R(D)$ is the central element of the theory and is one of the major contributions of Shannon [19]. It may be viewed as the upper bound on the performance for a compression system. In other words, it represents the lowest *information rate* R achievable while still maintaining a certain

distortion D or less [20]. Also, it implies that there is no coder that can sustain a level of distortion D at a rate below $R(D)$. Unfortunately, the $R(D)$ function does not reveal how such a coder can be constructed. Nonetheless, it can be employed in the design of compression systems. The $R(D)$ function itself has the property that it is convex, monotonically decreasing, and continuous. In general, the calculation of the $R(D)$ function is non-trivial. In fact, there are few cases for which analytical $R(D)$ functions are known. An extended description of the rate-distortion theorem is given in [1], [2].

2.3 Vector Quantization (VQ)

2.3.1 Introduction

Vector quantization (VQ) is a generalization of scalar quantization to the quantization of a vector (an ordered set of real numbers). VQ has received much attention as a powerful technique for data compression [3]. It has been shown in [4] that VQ is an effective method for coding speech, images, and video signals. Applications of VQ to such signals can be found in [8], [11], [21], [22], [23], [24].

According to Shannon's rate-distortion theory, a better performance is always achievable in theory by coding vectors instead of scalars, even though the data source is memoryless. Furthermore, the vector extension of Bennett's high quantization theory [25] shows explicitly the performance gain that may be achieved by

using VQ's. There exists major advantages of a VQ over a scalar quantizer. These advantages stem mainly from the VQ ability to:

- exploit both the linear and non-linear dependencies that may exist within an input vector.
- take advantage of its high dimensionality to generate non-cubic multidimensional partitions which provide better packing of the input space.
- track high order statistical characteristics of the input vector.

Despite the nice features of a VQ, it has some weakneses. According to rate-distortion theory, the lowest distortion that can be achieved is by using very large vector dimensions. However, an issue of recognized importance for the unconstrained VQ is that the size of the VQ codebook grows exponentially as a function of the vector size and the bit rate. As a result, both the computation and memory requirements associated with VQ will increase exponentially. Therefore, unconstrained exhaustive search VQ image coding has been limited to relatively small vector sizes such as 4×4 (16-dimensional).

2.3.2 Description of VQ

Vector Quantization is a method that maps a sequence of continuous or discrete vectors into a digital sequence suitable for storage or communication over a digital

channel [3]. It is a mapping Q of a k-dimensional Euclidean space, $\mathcal{R}^k$, into a finite subset $\mathcal{C} \subset \mathcal{R}^k$ .Thus,

$$Q : \mathcal{R}^k \mapsto \mathcal{C} \tag{2.4}$$

Moreover, in VQ, the source output is grouped into blocks or vectors $\boldsymbol{x}$ that belong $\mathcal{R}^k$. This vector of source output forms the input to the vector quantizer. At both the encoder and decoder of the vector quantizer, a set $\mathcal{C}$ of N-dimensional vectors called *codebook* of the vector quantizer exists. The vectors in this codebook, known as *codevectors*, are used to represent the vectors that have been generated from the source output, i.e, $\{\mathcal{C} = \boldsymbol{y}(1), \boldsymbol{y}(2), \cdots, \boldsymbol{y}(N)\}$. Each codevector $\boldsymbol{y}(j)$ is assigned a binary index $j \in \boldsymbol{J} \equiv 1, 2, \cdots, N$. At the encoder, the input vector is compared to each codevector in order to find the closest match.

The elements of this codevector are the quantized values of the source output. In order to inform the decoder about which codevector was found to be the closest to the input vector, the binary index of the codevector is transmitted or stored. A codebook identical to the one at the encoder is placed at the decoder in order to regenerate the codevector from its index [4] and [26]. Fig. 2.1 illustrates the VQ procedure.

Associated with every N-point VQ is a partition of $\mathcal{C}$ into regions or *cells*, where

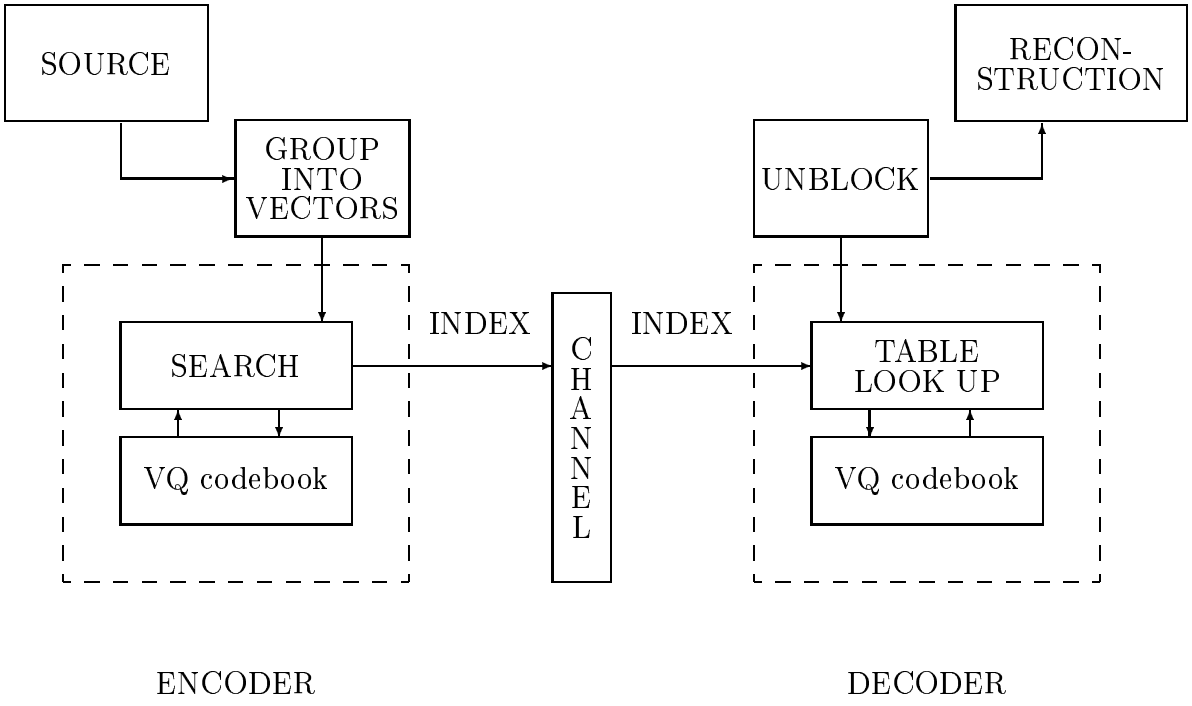


Figure 2.1: Vector Quantizer Procedure

the jth cell is defined by

$$\boldsymbol{S}(j) = \{\boldsymbol{x} \in \mathcal{R}^k : Q(\boldsymbol{x}) = \boldsymbol{y}(j)\} \tag{2.5}$$

Partitions of this type that are formed uniquely from the codebook and the squared error distortion are called *Voronoi Regions* or *Partitions*. The number of bits per sample used to address the contents of the codebook is $r = \frac{log_2 N}{k}$ and, in this case, the entropy is defined as

$$H = -\sum_{j=1}^{N} pr(\boldsymbol{x} \in \boldsymbol{S}(j)) \log_2 pr(\boldsymbol{x} \in \boldsymbol{S}(j)) \tag{2.6}$$

in bits per vector, where $pr(\boldsymbol{x} \in \boldsymbol{S}(j))$ denotes the probability that $\boldsymbol{x} \in \boldsymbol{S}(j)$.

Quantizing a vector $\boldsymbol{x}$ into $\boldsymbol{y}$ yields an error. In order to quantify that error (distortion), a measure $d(\boldsymbol{x}, \boldsymbol{y})$ between $\boldsymbol{x}$ and $\boldsymbol{y}$ is defined. By definition, a VQ is said to be optimal if the expected distortion

$$D = E[d(\boldsymbol{x}, \boldsymbol{y})] = \int d(\boldsymbol{x}, \boldsymbol{y}) f(\boldsymbol{x}) d\boldsymbol{x} \tag{2.7}$$

is minimized over all VQ's with N codevectors where $f(\boldsymbol{x})$ is the probability density function of the input [4]. There are two primary necessary conditions for optimality of VQ's [4]:

- Nearest Neighbor Condition: the quantizer mapping Q must be the *nearest-*

neighbor mapping given by

$$Q(\boldsymbol{x}) = \boldsymbol{y}(j), \quad \text{iff } d(\boldsymbol{x}, \boldsymbol{y}(j)) \leq d(\boldsymbol{x}, \boldsymbol{y}(i)) \quad \text{for } 1 \leq i \leq N \tag{2.8}$$

- Centroid Condition: each codevector $\boldsymbol{y}(j)$ be chosen so as to minimize the average distortion given a partition cell. In other words, $\boldsymbol{y}(j)$ is the vector $\boldsymbol{y}$ that minimizes the conditional distortion

$$D_j = E[d(\boldsymbol{x}, \boldsymbol{y}) | \boldsymbol{x} \in \boldsymbol{S}(j)] = \int_{\boldsymbol{x} \in \boldsymbol{S}(j)} d(\boldsymbol{x}, \boldsymbol{y}) f(\boldsymbol{x}) d\boldsymbol{x} \tag{2.9}$$

The vector $\boldsymbol{y}(j)$ is called the centroid of the cell $\boldsymbol{S}(j)$. The decoder of any VQ always performs a linear estimate of the input vector based on partial information about the input [4]. On the other hand, the optimal decoder for a given encoder is always an optimal linear estimate of the input vector. Therefore, the computation of the centroid for a particular cell depends on the distortion measure $d(\boldsymbol{x}, \boldsymbol{y})$ which is restricted to the squared error distortion. This is because it will result in a simple and explicit solution for the optimal codevector [4]. These optimality conditions are very important since they are frequently used as the basis for most of the VQ design algorithms.

2.3.3 VQ Design Techniques

The design of an optimal VQ from empirical data was extensively studied by Linde, Buzo, and Gray using a clustering approach referred to as the LBG algorithm [27]. The goal in designing an optimal VQ is to obtain a quantizer consisting of N code-vectors such that it minimizes the expected distortion given in 2.7. Optimality is said to be achieved if there is no other quantizer that can achieve the minimum expected distortion. Lloyd's basic algorithm for scalar quantization was extended to the general case of vector quantization. In this section, the LBG algorithm is presented following the formulation given in [27]. Let 2.7 be approximated by the time averaged square error distortion given by

$$D(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{N} \sum_{i=1}^{N} d(\boldsymbol{x}_i, \boldsymbol{y}(i)) \tag{2.10}$$

The algorithm for an unknown distribution training sequence is given as follows:

1. Let N be the number of levels and let $\epsilon \geq 0$ be the distortion threshold. Assume an initial N level codebook, say $\mathcal{C}_0$, and training sequence $(\boldsymbol{x}_j; j = 1, 2, \cdots, n)$ and m to be the number of iterations, initially to be zero.

2. Given $\mathcal{C}_m = (\boldsymbol{y}(j); j = 1, 2, \cdots, N)$, find the minimum distortion partition $\mathcal{P}(\mathcal{C}_m) = (\boldsymbol{S}(j); j = 1, 2, \cdots, N)$ of the training sequence: $\boldsymbol{x}_j \in \boldsymbol{S}(j)$ if

$d(\boldsymbol{x}_j, \boldsymbol{y}(i)) \leq d(\boldsymbol{x}_j, \boldsymbol{y}(i))$ for $1 \leq i \leq N$. Compute the average distortion

$$D_m = \frac{1}{n} \sum_{j=1}^{n} min_{\boldsymbol{y} \in \mathcal{C}_m} d(\boldsymbol{x}_j, \boldsymbol{y}) \tag{2.11}$$

3. If $\frac{(D_m - D_{m-1})}{D_m} \leq \epsilon$, stop the iteration with $\mathcal{C}_m$ as the final codebook; otherwise continue.

4. Find the optimal codebook $\boldsymbol{y}(\mathcal{P}(\mathcal{C}_m)) = (\boldsymbol{y}(\boldsymbol{S}(j); j = 1, 2, \cdots, N))$ for $\mathcal{P}(\mathcal{C}_m)$ where

$$\boldsymbol{y}(\boldsymbol{S}(j)) = \frac{1}{\|\boldsymbol{S}(j)\|} \sum_{i: \boldsymbol{x}_i \in \boldsymbol{S}(j)}^{m} \boldsymbol{x}_i \tag{2.12}$$

5. Set $\mathcal{C}_{m+1} = \boldsymbol{y}(\boldsymbol{S}(j))$, increment m to $m + 1$ and go to step 2.

There exist a number of techniques to obtain the initial codebook which is used as the seed for generating the codebook. The most widely used method is the splitting method by Linde *et al.* [27]. The splitting algorithm is based on calculating a centroid of the training sequence where it is split into two close vectors. The centroids or codevectors for the two partitions are then calculated. Each resulting vector is split into two vectors and the above procedure is repeated until an N level initial codebook is created. Splitting is performed by adding a fixed perturbation vector ϵ to each vector $\boldsymbol{y}(j)$ producing two vectors $\boldsymbol{y}(j) + \epsilon$, $\boldsymbol{y}(j) - \epsilon$. Another approach for designing initial codebooks is to employ product code techniques where a scalar quantizer is used k times successively to yield a k-dimensional VQ [4].

There exist a number of other techniques for designing codebooks such as Kohonen self-organization feature maps [4], [28], [29] and simulated annealing (Boltzman Machine) [4], [30].

2.3.4 Complexity of VQ

An issue of recognized importance for unconstrained VQ is that the size of the codebook grows exponentially as a function of the vector size k and the bit rate r. As a result, both the computation and memory requirements associated with VQ will increase exponentially. A codebook $\mathcal{C}$ consisting of N codevectors of dimension k requires $N = 2^{rk}$ vector distortion calculations to encode an input vector $\boldsymbol{x}$. Thus, even for simple distortion measures, the encoding complexity can quickly become unmanageable. For example, consider encoding an input vector representing a 4×4 block taken from a grayscaled image at a bit rate of 1.0 bits per pixel. Encoding on 4×4 input vector requires 65536 vector distortion calculations (more than one million scalar distortion calculations). Similarly, the memory required to store the codebook at both the encoder and decoder grows exponentially and is computed as $k2^{rk}$ pixels. In consequence, unconstrained exhaustive search VQ image coding has been limited to relatively small vector size such as 4×4.

2.3.5 Performance of VQ

While the performance of a VQ increases monotonically with the vector size k, the rate of increase is rather slow [31]. For example, Gallager [2] showed that the convergence rate to the $R(D)$ curve is approximately $\sqrt{\frac{\log_2 k}{k}}$. Berger in [1] reported that the convergence to the $R(D)$ curve with the vector size k can be *algebraic: linear, quadratic, etc.*, along at least on coordinate axis. VQ codes whose complexity is algebraic in vector size and whose performance is arbitrarily close to the $R(D)$ curve exist [1].

Practical VQ schemes have been limited to relatively small vector sizes. Hence, the complexity/performance tradeoffs of VQ are usually not good. In addition, transmission or storage noise can result in significant degradations in the VQ performance. Also, to obtain a robust VQ design, a very large training set is usually needed. Finally, while practical VQ's may code images with acceptable subjective quality at moderate bit rates, at low rates the quality is generally poor with visible block artifacts.

2.4 Constrained VQ

Direct use of VQ suffers from a serious complexity barrier that greatly limits its practical use as a complete and self-contained coding technique. Unconstrained VQ is severely limited to rather modest vector dimension and codebook sizes for practi-

cal problems. Several techniques has been developed which apply several constraints to the structure of the VQ codebook and yield a correspondingly altered encoding algorithm and design technique. As a result, higher vector dimensions and larger codebook sizes become feasible [4]. In general, these methods compromise the performance achievable with unconstrained VQ. However, they provide very useful and favorable trade-offs between performance and complexity. In other words, it is often possible to reduce the complexity by orders of magnitude while paying only a slight penalty in average distortion. Moreover, the constrained VQ's can often be designed for larger dimensions and rates. Therefore, quality, which is simply not possible for unconstrained VQ, becomes practicable and hence no performance is "lost" at all [4].

One approach to mitigate the complexity barrier is to impose a certain structural constraints on the codebook. In such case, the codevectors cannot be arbitrary located as points in the k-dimensional space but are distributed in a restricted manner which allows a much easier search for the nearest neighbor. Examples of such constrained codebook structures are Tree-Structured VQ (TSVQ) [3], [22], [32], Lattice VQ [33], [34], Product-Code VQ (PCVQ) [23], [35] , Multi-Stage or Residual VQ (RVQ) [4]. The TSVQ, PCVQ, Lattice VQ, and RVQ are briefly described in the next coming sections. RVQ is also disscused in next chapter in a more detailed manner since it is the focus of this work. RVQ has the ability to reduce the VQ complexity substantially in addition to its good complexity/performance trade-offs.

2.4.1 Tree-Structured VQ

The most popular computational complexity reduction technique is tree-structured VQ (TRVQ) [3], [22]. TSVQ was first introduced by Buzo *et al* [22] with a special case of uniform binary trees. Extensions to more general trees where given in [5] and [36]. Intensive research was done on TSVQ. Fixed rate, variable rate, uniform, nonuniform, constrained, and unconstrained trees, were extensively studied in [21], [32], [37]. In such studies, good speech and image compression results were reported. The main advantages of TSVQ's are in its suitability for progressive transmission, it is relatively low sensitiv to channel noise, and ease of use as a component of a variable rate compression system.

In TSVQ, a sequence of binary (or larger) searches is performed instead of large search as in full search VQ. As a result, its encoding complexity increases as $\log N$ instead of N where N is the number of codevectors (nodes of the bottom layer) of the tree. Considering a binary TSVQ, the search is done in the following way: starting at the root node of the tree, if the input vector is closer in a minimum distortion sense to the left child, transmit a '0' and descend to the left child. If it is closer to the right child, transmit a '1' and descend to the right child. Repeat with the selected child and continue until a leaf of the tree is reached. If the leaves all lie at the same depth, then a fixed rate code is generated.

The performance of a TSVQ, in general, suffers some degradation compared to

the performance of full search VQ with the same number of codevectors. This is due to its suboptimal design procedure as well as the constraint on the search [37]. In addition, the memory requirements of TSVQ at both encoder and decoder are nearly doubled.

2.4.2 Lattice VQ

A special class of vector quantizers that are of particular interest because of their highly regular structure are called lattice quantizers [4]. The quantizer here form a *regular lattice* in the k-dimensional space. A k-dimensional *nondegenerate* lattice L has codevectors $\boldsymbol{y}$ that satisfy

$$\boldsymbol{y} = \sum_{i=1}^{k} l_i \boldsymbol{u}_i$$

where $\{l_i; i = 1, 2, \cdots, k\}$ are integer variable and $\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_k$ are k linearly independent vectors in $\mathcal{R}^k$. These vectors are called the *generators* for the lattice.

Except for uniformly distributed sources, lattice VQ's are suboptimal in the sense of minimizing the average distortion. This is dut to their regular structure. Lattice VQ has poor performance at low bit rates, e.g, rates used in image coding. However, they usually work well for high bit rate and low distortion applications [4].

2.4.3 Product Code VQ

An age-old technique for handling an unmanageably large task is to decompose it into smaller sub-tasks. Essentially, this is the idea of product codes (PCVQ). In VQ, the task that can often become too large is encoding. One way to reduce the task is to partition the vector into subvectors, each of lower dimensionality. A VQ codebook is then designed for each subvector and used to generate an index. The combination of indices from the VQ codebooks is used by the product VQ decoder to specify the coded vector. The obvious reduction in complexities and memory requirements , however, comes with a price. There exists typically statistical dependence between the different subvectors. By separate coding of each subvector, the possibility of exploiting that dependence will be lost. Therefore, this leads to a significant degradation in rate-distortion performance [4]. There are many ways that can compensate for this degradation. For example, by finding a method that will produce subvectors which are almost statistically independent, one can increase the $R(D)$ performance while still maintain a relatively low complexity. On the other hand, the PCVQ codebooks can be jointly optimized which will lead to improvement in the performance at the expense of some additional design complexity. Mean-residual VQ [36], Gain-shape VQ [38], [39], reflected product code VQ [36], and Residual VQ (RVQ) are all examples of PCVQ's.

2.4.4 Residual VQ

One alternative technique that has proved valuable in a number of speech and image coding applications is *multi-stage* or *cascaded* VQ [4]. This method is also referred to as *residual* VQ (RVQ). RVQ is one of the simple and efficient types of structurally-constrained VQ designs. A P-stage k-dimensional RVQ encompass a wide range of quantizers. Particularly, there are three well-known quantizers which are special cases of RVQ. When $P = 1$, we have the conventional VQ. When $k = 1$, we will have what is called the multi-stage (residual) scalar quantizer. When both $P = 1$ and $k = 1$, the RVQ in this case is the Lloyd-Max scalar quantizer [4],[27].

The basic idea of RVQ is to divide the encoding task into successive stages, where the first stage performs a relatively crude quantization of the input vector using small codebook. Then, a second stage quantizer operates on the error vector between the original and quantized first stage output. The quantized error vector then provides a second approximation to the original input vector thereby leading to an accurate representation of the input. A third stage quantizer may then be used to quantize the second stage error vector to provide a further refinement and so on.

RVQ is a design that is capable of reducing both memory and computation costs. In addition, it can operate over a large range of bit rates and vector dimensions. A simple two-stage RVQ was proposed by Juang and Gray in 1982 for coding of speech

signals [9]. This RVQ encoder utilizes a computationally inexpensive, but suboptimal sequential single-path search. The performance of this computationally cheap RVQ was found to degrade significantly as the number of stages grows beyond two. This degradation can be interpreted as follows. In their method, each stage codebook is generated while considering only the error due to previous stages; the error due to subsequent stages is ignored. In 1989, Barnes and Frost [10], introduced a jointly optimized RVQ or simply RVQ design. In their design, an attempt was made to minimize the overall quantization error of the RVQ in lieu of merely optimizing the individual stages in isolation. They demonstrated that a sequential single-path search through VQ stages cannot, in general, utilize all the available codevectors. They employed *M*-search, an efficient multi-path tree search algorithm, to search the stage codebooks. The design resulted in improvement. However, the increase in performance comes at the expense of additional computations.

2.5 Distortion Measures

Ideally a distortion measure should be tractable to permit analysis and design. In addition to that, it should be computable so that it can be evaluated in real time for guiding the actual encoding process for encoders which select the minimum distortion output. Also, the distortion measure should be subjectively meaningful so that large or small average distortion values correlate with bad and good subjective

quality as perceived by the ultimate user of the codevector sequence [4].

The most convenient and widely used measure of distortion between the input vector $\boldsymbol{x}$ and the quantized vector $\boldsymbol{y}$ is the *squared error* or the *squared Euclidean distance* between the two vectors. The squared error is defined as

$$\begin{aligned} d(\boldsymbol{x}, \boldsymbol{y}) &= \|\boldsymbol{x} - \boldsymbol{y}\|^2 \\ &= (\boldsymbol{x}, \boldsymbol{y})^t (\boldsymbol{x}, \boldsymbol{y}) \\ &= \sum_{i=1}^{k} (\boldsymbol{x}_i - \boldsymbol{y}(i))^2 \end{aligned} \tag{2.13}$$

where t is the transpose of the vector. The *average squared error distortion* or the *average distortion* is defined as

$$D = E[d(\boldsymbol{x}, \boldsymbol{y})] = E(\|\boldsymbol{x} - \boldsymbol{y}\|^2) \tag{2.14}$$

Basically, this measure is frequently associated with energy or power of an error signal and therefore has some intuitive appeal in addition to being an analytically tractable measure for many reasons.

Numerous alternative distortion measures exist. One of the common measures

is the Holder norm or the l_v-norm defined as

$$\begin{aligned} d(\boldsymbol{x},\boldsymbol{y}) &\stackrel{\triangle}{=} \|\boldsymbol{x}-\boldsymbol{y}\|_v \\ &= \{\sum_{i=1}^{k} |\boldsymbol{x}_i - \boldsymbol{y}(i)|^v\}^{1/v} \end{aligned} \tag{2.15}$$

One can notice that Eq. 2.15 depends on the power $v \in \mathcal{R}$ ($\mathcal{R}$ is the set of real numbers). This measure is still useful since it is a distance and hence will satisfy the triangular inequality [27]. Other distortion measures are the l_∞ or Minkowski norm,

$$d(\boldsymbol{x},\boldsymbol{y}) = max_{1\le i\le k}|\boldsymbol{x}_i - \boldsymbol{y}(i)| \tag{2.16}$$

the weighted-squares distortion,

$$d(\boldsymbol{x},\boldsymbol{y}) = \sum_{i=1}^{k} w_i|\boldsymbol{x}_i - \boldsymbol{y}(i)|^2 \tag{2.17}$$

where $w_i \ge 0; i = 1, \cdots, k$, and the more general quadratic distortion

$$d(\boldsymbol{x},\boldsymbol{y}) = (\boldsymbol{x}-\boldsymbol{y})\boldsymbol{B}(\boldsymbol{x}-\boldsymbol{y})^t \tag{2.18}$$

where $\boldsymbol{B}$ is a $k \times k$ positive definite matrix [27].

All the distortion measures introduced here depend on the difference (error) between the input and the quantized vectors. Thus, they are called difference dis-

tortion measures. There are other distortion measures in the literature that are more complicated than the difference distortion measures and are normally used for speech compression systems such as the distortion method of Itakura and Saito, and Chaffee [27]. However, in image coding, difference distortion measures, particularly the squared error, are extensively used in the literature.

In order to allow fair comparisons between the method of coding developed in this thesis and those reported in the literature, the following procedure is adopted:

1. Test images that are *not* part of the training set are used, and

2. Objective performance measure like the signal-to-quantization noise ratio (SNR) or peak signal-to-quantization noise ratio (PSNR) is provided.

The objective performance measure used in all synthetic coding experiments was the SNR which is defined by

$$SNR = -10 \log_{10} \frac{\sum_{i=1}^{N} (x(i) - \hat{x}(i))^2}{\sum_{i=1}^{N} (x(i))^2} \tag{2.19}$$

where N is the number of samples in the sequence. $x(i)$ and $\hat{x}(i)$ represent the original and the coded values, respectively, of the ith sample. One the other hand, the objective performance measure used in all image coding experiments was the

PSNR. The PSNR is defined as

$$PSNR = -10 \log_{10} \frac{\sum_{i=1}^{N} \sum_{j=1}^{N} (x(i,j) - \hat{x}(i,j))^2}{(N^2)(255)^2} \tag{2.20}$$

where $N \times N$ is the size of the image and $x(i,j)$ and $\hat{x}(i,j)$ represent the original and coded values, respectively, at the ith row and the jth column.

2.6 Image Sources

An image, whether it is still or moving, monochrome or color, is just a signal to be compressed. However, the compression technique used usually depends on the type of the image signal being compressed. To compress video frames, for example, a good compression method should exploit both intra-frame and inter-frame statistical dependencies.

Input images associated with image processing systems can be acquired in a number of ways. Common input devices include analog and digital cameras, image and document scanners, and line scanners. Depending on the application, image data may be collected using video cameras, CT scanners, radio telescopes, etc [17]. Several techniques exist for generating and displaying image data, however, our focus is on dealing with the data itself.

An image signal is represented as an n-dimensional array of picture elements *pixels* [40]. When $n = 2$, it means that we have a still image, $n = 3$ represents a

monochrome video sequence, a medical image sequence or a still color image, etc. Hence, by treating images as an n-dimensional arrays of correlated random variables, we are able to develop compression schemes that can suite different types of imagery.

The compression technique studied here can be extended to image sequences such as video and CT image scans whether they are monochromed or colored. However, the focus of the work is mainly on gray-scaled monochrome still images. These images are considered to be standard for research which are available in the University of Southern California (USC) image database (www.sipi.usc.edu). The pixel values of such images are of 256 gray levels from 0 to 255 and stored in one-byte format. The famous LENA image is extensively used in this thesis because it is widely used in the related literature, which makes it convenient for comparison purposes.

Chapter 3

Fixed Rate Reflected Residual Vector Quantization

3.1 Some Preliminaries

3.1.1 A Review of Residual Vector Quantizer Structure

Let $\boldsymbol{X}_1$ be an n-dimensional random vector described by the probability distribution function $F_{\boldsymbol{X}_1}(\cdot)$, where $\boldsymbol{x}_1$ is a particular realization of the random vector $\boldsymbol{X}_1$. A P-stage residual vector quantizer (RVQ) of $\boldsymbol{X}_1$ consists of a finite sequence of P vector quantizers $(\mathcal{C}_p, Q_p, \mathcal{P}_p); 1 \leq p \leq P$, each of which quantizes the residual $\boldsymbol{x}_p$ of the preceding stage $(\mathcal{C}_{p-1}, Q_{p-1}, \mathcal{P}_{p-1}); 1 \leq p \leq P$. The pth stage codebook $\mathcal{C}_p$ consists of N_p number of stage code vectors. The codevectors are indexed by the subscript

p; that is, $\mathcal{C}_p = \{\boldsymbol{y}_p(1), \boldsymbol{y}_p(2), \cdots, \boldsymbol{y}_p(N_p)\}$. Each pth stage codebook is associated with a set of partition cells; that is, $\mathcal{P}_p = \{S_p(1), S_p(2), \cdots, S_p(N_p)\}$. The map Q_p applied to the pth stage input $\boldsymbol{x}_p$ yields $Q_p(\boldsymbol{x}_p) = \boldsymbol{y}_p(j_p)$, if and only if $\boldsymbol{x}^p \in S^p(j_p)$.

In practice, each $Q_p(\cdot)$ is realized as a composition of encoder map $\mathcal{E}(\cdot)$ and a decoder map $\mathcal{D}(\cdot)$. The pth encoder map is defined as $\mathcal{E}_p(\boldsymbol{x}_p) = j_p$, if and only if $\boldsymbol{x}_p \in S_p(j_p)$. A 2-stage RVQ is shown in Fig. 3.1. The indexes produced by the sequence of stage encoder maps are concatenated to form an index P-tuple $\boldsymbol{j} = (j_1, j_2, \cdots, j_P)$. Each P-tuple is a *product codeword* and is an element of the stage index sets $\boldsymbol{J} \in \{J_1 \times J_2 \times \cdots J_P\}$. The decoder maps each stage index j_p to the corresponding code vector $\boldsymbol{y}_p(j_p)$. The quantized representation $\hat{\boldsymbol{x}}_1$ of the input source vector $\boldsymbol{x}_1$ is formed by the sum of the selected stage code vectors,

$$\hat{\boldsymbol{x}}_1 = \sum_{p=1}^{P} \boldsymbol{y}_p(j_p). \tag{3.1}$$

3.1.2 Direct Sum Quantizers

For the analysis purpose, the notion of an equivelant single-stage quantizer is introduced for a multistage RVQ, referred to as direct sum quantizer. The direct sum single-stage quantizer produces the same representation of the source input $\boldsymbol{x}_1$ as does the corresponding multistage RVQ. A direct sum quantizer is specified by a

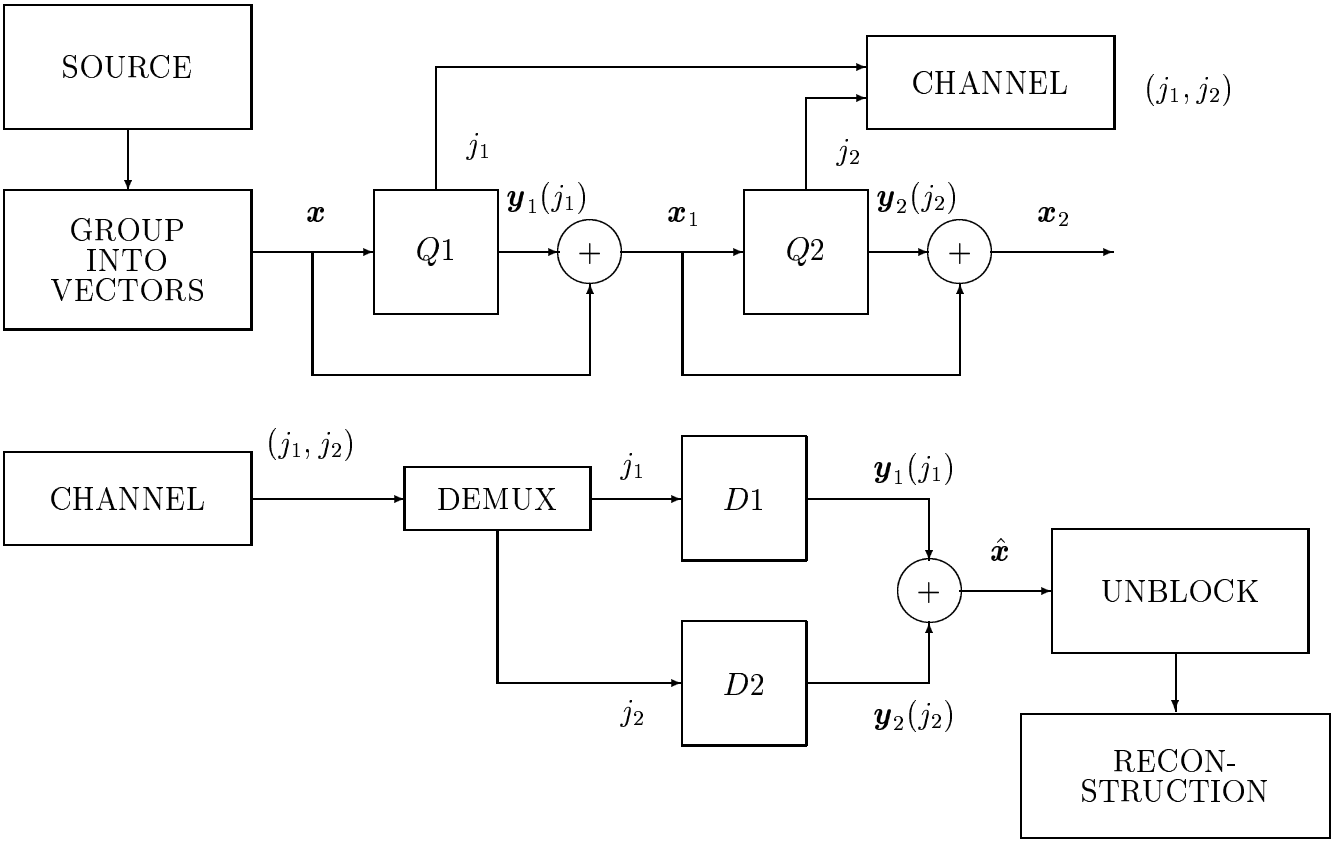


Figure 3.1: A two-stage Residual Vector Quantizer

triple $(\mathcal{C}_e, Q_e, \mathcal{P}_e)$ consisting of a *direct sum codebook*, *direct sum mapping*, and *direct sum partition*, respectively. The direct sum codebook $\mathcal{C}_e$ is a set of all possible sums of stage code vectors; that is, $\mathcal{C}_e = \mathcal{C}_1 \oplus \mathcal{C}_2 \oplus \cdots \oplus \mathcal{C}_P$. There are $N_e = \prod_{p=1}^{P} N_p$ direct sum code vectors in $\mathcal{C}_e$. The direct sum codevectors $\boldsymbol{y}_e(\cdot)$ are indexed by the P-tuples $\boldsymbol{j} = (j_1, j_2, \cdots, j_P)$. The $S_e(\boldsymbol{j})$ is the partition cell corresponding to the $\boldsymbol{j}$th direct sum code vector $\boldsymbol{y}_e(\boldsymbol{j})$ according to nearest neighbor rule. The *direct sum partition* $\mathcal{P}_e$ is the collection of all direct sum partition cells. The *direct sum mapping* will represent source input $\boldsymbol{x}_1$ with direct sum code vector $\boldsymbol{y}_e(\boldsymbol{j})$ if and only if $\boldsymbol{x}_1 \in S_e(\boldsymbol{j})$.

Direct sum codebook sets have specific tree structures that are associated with each permutation of the stage codebooks. Each direct sum codevector $\boldsymbol{y}_e(\cdot)$ is interpreted as the terminating node of a tree branch. Fig. 3.2 illustrates the tree structure associated with a scalar, three stage, binary (two code vectors/stage) residual quantizer. The root node corresponds to the origin of the space in which the code vectors lie, each additional level of the tree represents a particular residual stage. The nodes at the first level below the root node are the values of the first stage codevectors. The second level node values are the direct sum code vectors of the first and second stage codebooks. Similarly, the third level node values corresponds to the direct sum code vectors of the first, second, and third stage codebooks.

3.1.3 Entanglements

Consider two scaler direct sum code trees, each one represents a 3-stage binary residual quantizer, as illustrated in Fig. 3.2 and 3.3. Note the labeling of the direct sum partition cells $S_e(\cdot)$ produced by a sequentional encoder which makes nearest neighbor decisions between the $\boldsymbol{y}_p(j_p)$. The labeling of $\boldsymbol{y}_e(\cdot)$ and $S_e(\cdot)$ is consistent for the second tree, that is, $\boldsymbol{y}_e(\boldsymbol{j}) \in S_e(\boldsymbol{j})$, but not for the second tree. In particular, $\boldsymbol{y}_e(5) \in S_e(3)$, $\boldsymbol{y}_e(3) \in S_e(4)$, $\boldsymbol{y}_e(6) \in S_e(5)$ and $\boldsymbol{y}_e(4) \in S_e(6)$. The branches of the tree structure shown in Fig. 3.2 are intertwined and thus results in *entangled* tree.

The entanglement of an RVQ tree impacts the computational efficiency of optimal sequential encoders. It is not difficult to imagine the case where the complexity of an optimal sequential encoder might be very complex.

There are generally two approaches to deal with an entangled RVQ tree structures. The first approach is to use entanglement-permissive encoders. In this respect M-search technique has been shown by many authors to be an effective search algorithm for an entangled tree [7], [8]. In M-search encoding, the first-stage encoder finds the M closet first-stage code vectors. The second-stage codebook is then searched for each of the M residual vectors output by the first stage. The M best two-stage path segments selected from the restricted set of examined first- and second-stage code vector combinations are retained as candidate survivor paths through the RVQ tree structure. This process is repeated on the next stage, and so

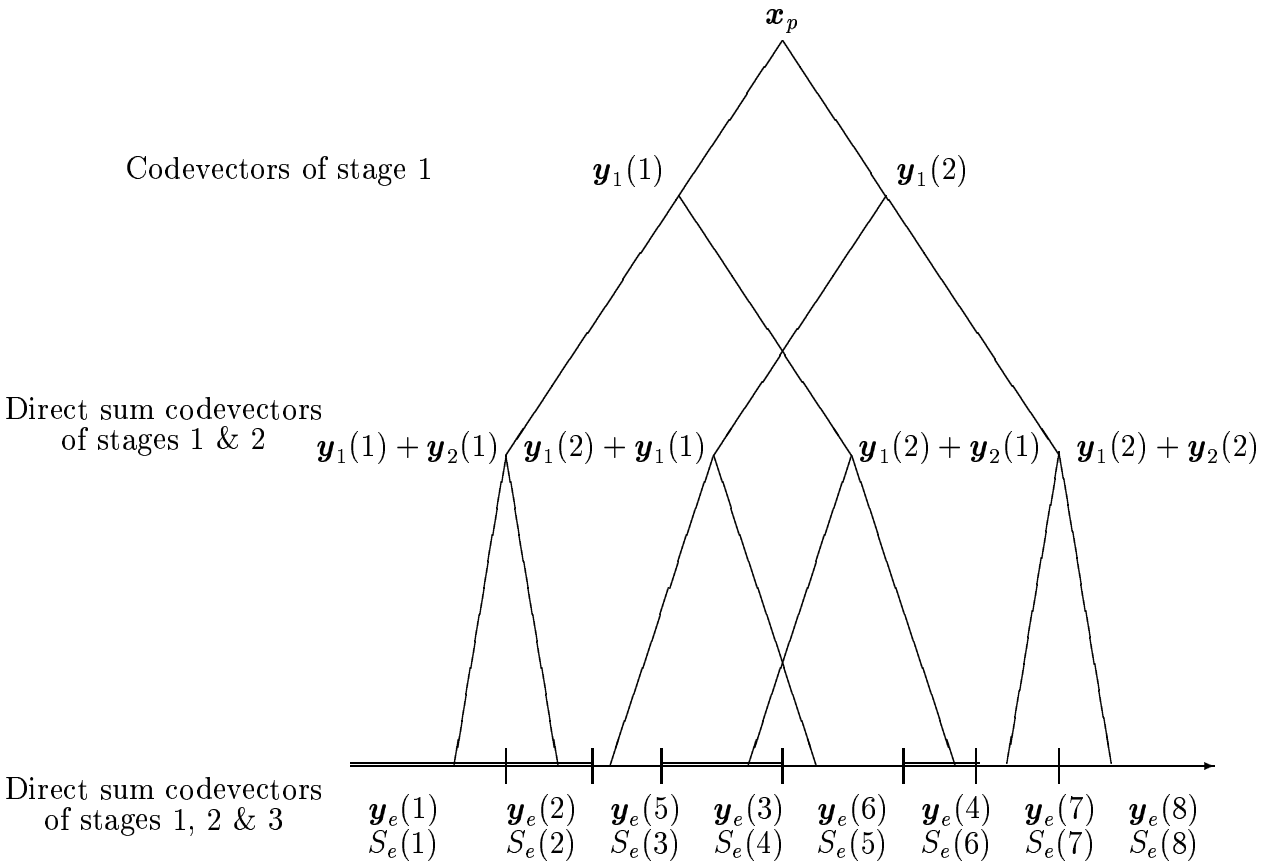


Figure 3.2: Three-stage scalar binary RVQ tree structure

on, until the last RVQ stage has been searched, where the best of the M complete candidate paths are retained. Direct sum codebooks were designed and implemented for synthetic data [7] and imagery [8]. An entropy-constrained RVQ design was also proposed, based on the M-search algorithm with Lagrangian formulation [12]. The second approach to tackle entangled tree structures is described in the next subsection.

3.2 Reflected RVQ (RRVQ)

If the tree structure is un-entangled, then the optimal Voronoi partitions form connected sets and there exists a single-path search that may pose as optimal and efficient RVQ tree search. To look more closely into this possibility, we wish to make explicit the relationship between the direct sum partition $\mathcal{P}_e$ and stagewise partition $\mathcal{P}_p$ for $1 \leq p \leq P$.

Let us first find the optimal Voronoi partition $\mathcal{P}_2$ for the second residual stage. A tree structure of three stages, two codevectors/stage RVQ is illustrated in Fig. 3.3. The root node of the tree represents $\boldsymbol{x}$. The leaf nodes represents the set $\mathcal{C}_e$ of the direct sum codevectors. The intermediate nodes represent partial sums of the direct sum codevectors and the branches represent stage codevectors.

For this purpose, we assume the first-stage quantizer has made the optimal decision and thus fix $\boldsymbol{y}_1(j_1)$. The second-stage optimal Voronoi cell $S_2(j_2)$ consists of

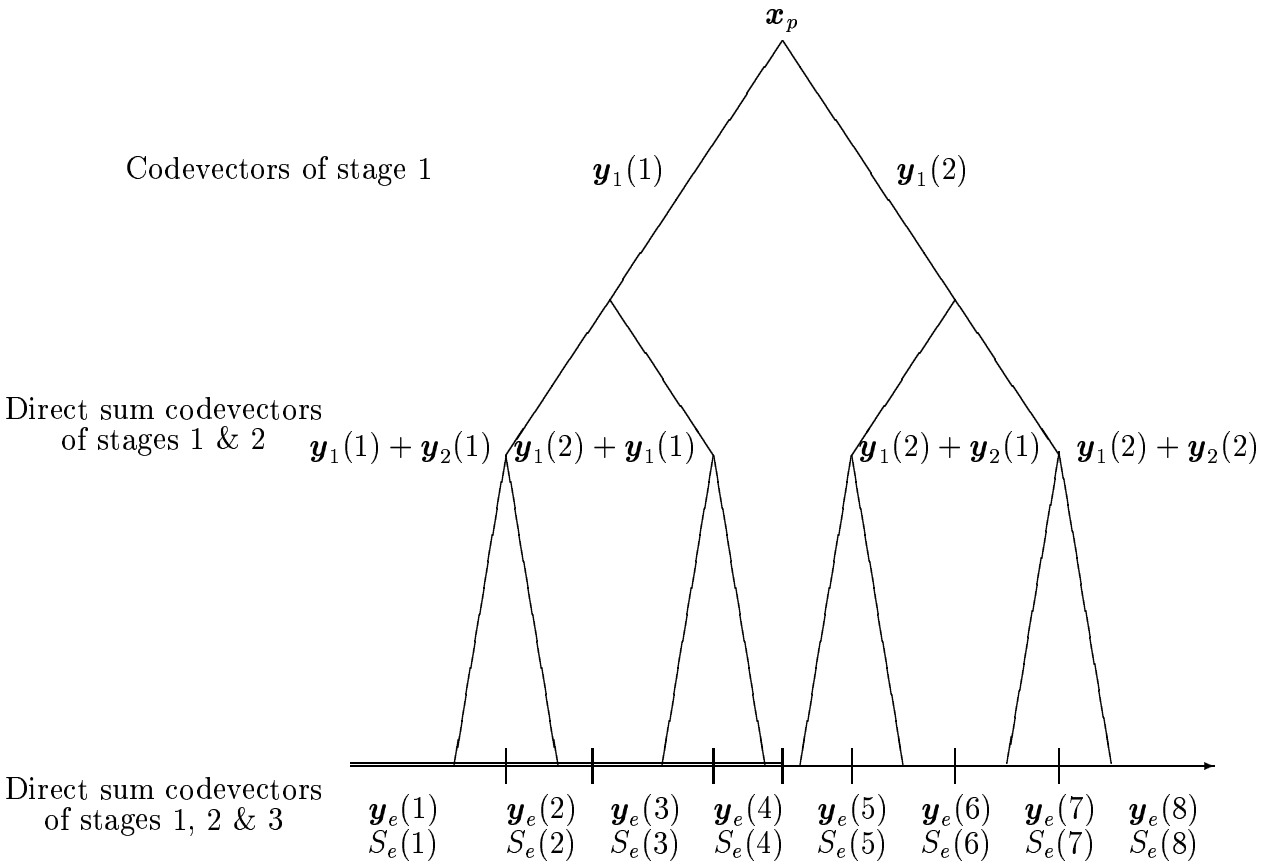


Figure 3.3: Three-stage scalar RRVQ tree structure

all residual vectors $\boldsymbol{x}_2 = \boldsymbol{x}_1 - \boldsymbol{y}_1(j_1)$ that satisfy the following relationship

$$d(\boldsymbol{x}_2, \boldsymbol{y}_2(j_2) + \sum_{p=3}^{P-1} \boldsymbol{y}_p(k)) \leq d(\boldsymbol{x}_2, \sum_{p=2}^{P-1} \boldsymbol{y}_p(k)) \tag{3.2}$$

Eq. 3.2 identifies $S_p(j_p)$ as the subset of $\mathcal{R}$ that is Voronoi (nearest neighbor) with respect to the terminating nodes of all paths of the tree $\{\mathcal{C}_p \oplus \mathcal{C}_{p+1} \oplus \cdots \oplus \mathcal{C}_P\}$ which contains $\boldsymbol{y}_p(j_p)$ in their construction.

One may deduce that in order to guarantee the construction of a fully unentangled tree, we need to make sure the optimal stagewise Voronoi cells are connected. However, these are necessary but not sufficient conditions for the residual vector quantization. In case of vector quantization, although the stagewise Voronoi cells $S_p(j_p)$ may be connected, it has the shape of a multi-faceted polytope. The boundaries of such Voronoi cells are complex. However, if we allow only two code vectors $\{\boldsymbol{y}_p(1), \boldsymbol{y}_p(2)\}$ at each stage, then the Voronoi boundary between $S_p(1), S_p(2)$ is a simple hyperplane. To obtain such a structure, we impose an additional *reflection* constraint on the RVQ tree. This constraint dictates that each terminating node, belonging to one parent code vector and adjacent to the hyperplane boundary, must have a matching node that originates from the other parent code vector.

If two code vectors $\{\boldsymbol{y}_p(1), \boldsymbol{y}_p(2)\}$ are allowed in a given stage, then the Voronoi boundary is a plane of equal distortion between two code vectors. This boundary at any given residual stage p can be specified by a midway point $\boldsymbol{m}_p$ between the

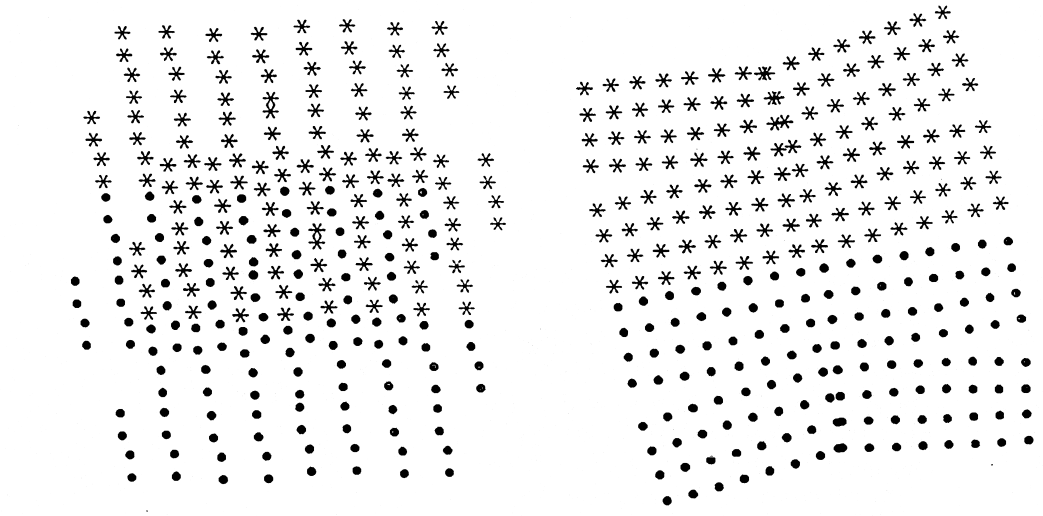
two given code vectors, given by

$$\boldsymbol{m}_p = \frac{1}{2}\{\boldsymbol{y}_p(1) + \boldsymbol{y}_p(2)\} \tag{3.3}$$

The normal vector at a given stage p, $\boldsymbol{n}_p$, is defined to be the line joining two code vectors $\overline{\boldsymbol{y}_p(1)\boldsymbol{y}_p(2)}$. The equation of the plane through the midway point $\boldsymbol{m}_p$ perpendicular to $\boldsymbol{n}_p$ is

$$\boldsymbol{n}_p \cdot \overline{\boldsymbol{m}_p \boldsymbol{z}_p} = 0 \tag{3.4}$$

where $\boldsymbol{z}_p$ is a point in the plane. In order for this hyperplane to specify the boundary between adjacent children of the two code vectors, the input vectors of the *pth*-stage are reflected to one side of the hyperplane boundary, and by convention, all $\boldsymbol{x}_p$ which belong to Voronoi cell $S_p(2)$ are reflected to second Voronoi cell $S_p(1)$. This results in the second Voronoi cell to be empty and the first Voronoi cell containing all the residuals $\boldsymbol{x}_p$. Next, we subtract $\boldsymbol{y}_p(1)$ from all the $\boldsymbol{x}_p$'s that are now residing in the first Voronoi cell. Then, residual vectors that represent the next stage code vectors, will lie in the *reflected residual space*. If we are to unreflect all the reflected stage codebooks, the resulting direct sum codebook has the desired symmetry properties. To illustrate the structure of RRVQ codebook, Fig. 3.4 (a) shows the codevector constellation for (8-stages, two codevectors/stage) two dimensional RVQ designed for Gaussian source. Similarly, Fig. 3.4 (b) shows the codevector constellation



(a) (b)

Figure 3.4: Gaussian source coded with a binary 8-stage, two-dimensional quantizer. (a) Equivalent code vector constellation of RVQ. (b) Code vector constellation of RRVQ.

for (8-stages, two codevectors/stage) two dimensional RRVQ designed for Gaussian source. All the direct sum codevectors that involve first codevector of first RVQ stage in their construction are represented as dots. On the other hand, asterisks are used for the direct sum codevectors constructed with second code vector of first stage RVQ. Fig. 3.4 (a) indicates severe codevector diffusion for RVQ, whereas the RRVQ (Fig. 3.4 (b)) shows no diffusion and hyperplane boundaries are evident.

Rate-distortion results of comparative experiments for the RRVQ are reported in [10]. For the memoryless Gaussian source, at rates 0.5 and 1.0 bits/sample, the RRVQ performance is not much better than Juang's and Gray's RVQ. For the Gauss-Markov and imagery sources, the RRVQ performance is upper bounded by the performance of the exhaustive search RVQs and lower bounded by the performance of the Juang Gray RVQs. This displays a rather disappointing picture of the performance of fixed-rate RRVQ.

The prior discussion has shown that the imposition of a reflection constraint leads to an unavoidable increase in distortion, but since structured systems are inherently *less random* or *more ordered*, it is reasonable to expect that the imposition of structure also reduced output entropy. Reduction in entropy is shown to partially compensate for the increase in distortion. This notion was tested on other structured codebooks and have shown that indeed this is the case. Here our intention is to extract such gains from the reflected RVQ.

3.3 The Fixed Rate Design Algorithm

The RRVQ encoding design algorithm is described as follows [10]:

1. Given a training set, number of stages P, a distortion thresholds $\epsilon \geq 0$ and $\epsilon^d \geq 0$, and a distortion measure $d(\cdot, \cdot)$, initialize the number of steps m to be -1, $P_m = 0$ and $D_m = \inf$.

2. If $P_m = P$, then $(\mathcal{C}_m^1, \mathcal{C}_m^2, \cdots, \mathcal{C}_m^P)$ will be the final stage codebooks. Otherwise, continue.

3. Let $P_m = P_m + 1$, and $m = m + 1$. For the mth stage of the quantizer, select two different vectors (you can use the splitting algorithm).

4. Determine the mid points and normal vectors for $1 \leq p \leq P_m$.

5. Partition the training set into the sequence of optimal stagewise partitions via the following steps:

   - For each training set vector $(\boldsymbol{x}_j); 1 \leq j \leq n$:

     - For each stage $1 \leq p \leq P_m$:

       * If $d(\boldsymbol{x}_p(j), \boldsymbol{y}_p(1)) \leq d(\boldsymbol{x}_p(1), \boldsymbol{y}_p(2))$, then assign $\boldsymbol{x}_p(j)$ to $\boldsymbol{S}_p(1)$ and form the unreflected residual vector $\boldsymbol{x}_{p+1}(j) = \boldsymbol{x}_p(j) - \boldsymbol{y}_p(1)$.

       * If $d(\boldsymbol{x}_p(j), \boldsymbol{y}_p(1)) > d(\boldsymbol{x}_p(1), \boldsymbol{y}_p(2))$, then assign $\boldsymbol{x}_p(j)$ to $\boldsymbol{S}_p(2)$. Calculate reflection distance $d = |\boldsymbol{n}_p \cdot \overline{\boldsymbol{m}_p \boldsymbol{x}_p(j)}|$ and reflect

6. Compute the average distortion D_m from the total residual error resulting from step number 5.

$$D_m = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_{P_m}(i) \tag{3.5}$$

7. If $(D_m - D_{m-1})/D_m \leq \epsilon$, then go to step 3. Otherwise continue.

3.4 Performance of Fixed Rate RRVQ

Simulations were performed for zero-mean unit variance Gaussian and Laplacian data. The probability density function (PDF) for the used Gaussian random variable is given by

$$f_X(x) = \frac{1}{\sqrt{2\pi}} exp(\frac{-x^2}{2}) \tag{3.6}$$

The $R(D)$ function for such random variable using the mean-square-error as a distortion measurement is given by[1], [2]:

$$R(D) = \frac{1}{2} max(0, -\log D) \tag{3.7}$$

On the other hand, the PDF for the Laplacian data used in the simulation is given as

$$f_X(x) = \frac{1}{\sqrt{2}} exp(-\sqrt{2}|x|) \tag{3.8}$$

Although there is no $R(D)$ function available for this source using the mean-square-error, the $R(D)$ values at some typical bit rates are known [1].

Training data were made of 2,000,000 samples. The resulting codebooks were tested for 100,000 samples outside the training set. Both RVQ and RRVQ were simulated for a two-dimensional vector source. Comparisons were made between several rates starting form 0.5 bits per sample (bps) to 4.0 bps for RVQ (M=4 and 2-codevectors per stage) and RRVQ. Fig. 3.5 and Fig. 3.6 indicate that RVQ outperforms RRVQ for both Gaussian and Laplacian sources, respectively. The difference in SNR between RVQ and RRVQ was 0.5-dB on average. This is due to the fact that RRVQ has added constraint of reflection on its direct sum codebook as compared to the RVQ direct sum codebook.

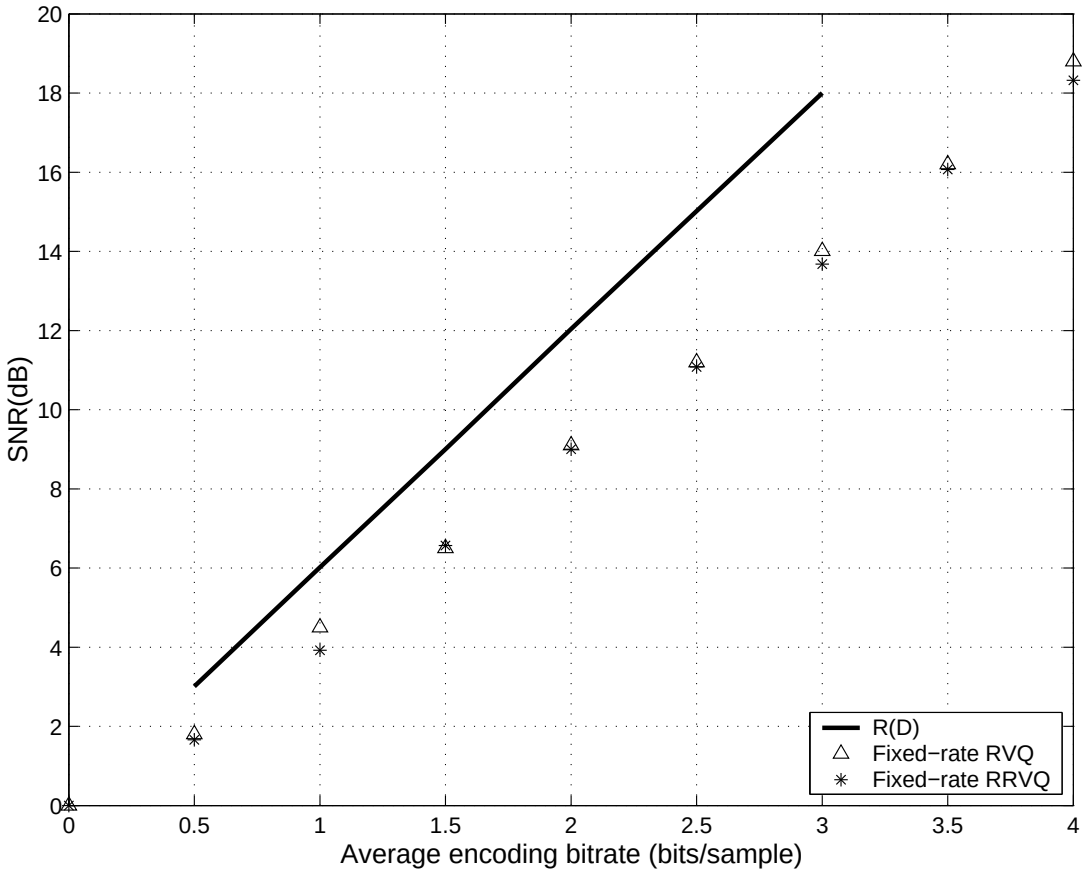


Figure 3.5: Performance comparison of RRVQ with RVQ for the memoryless Gaussian source

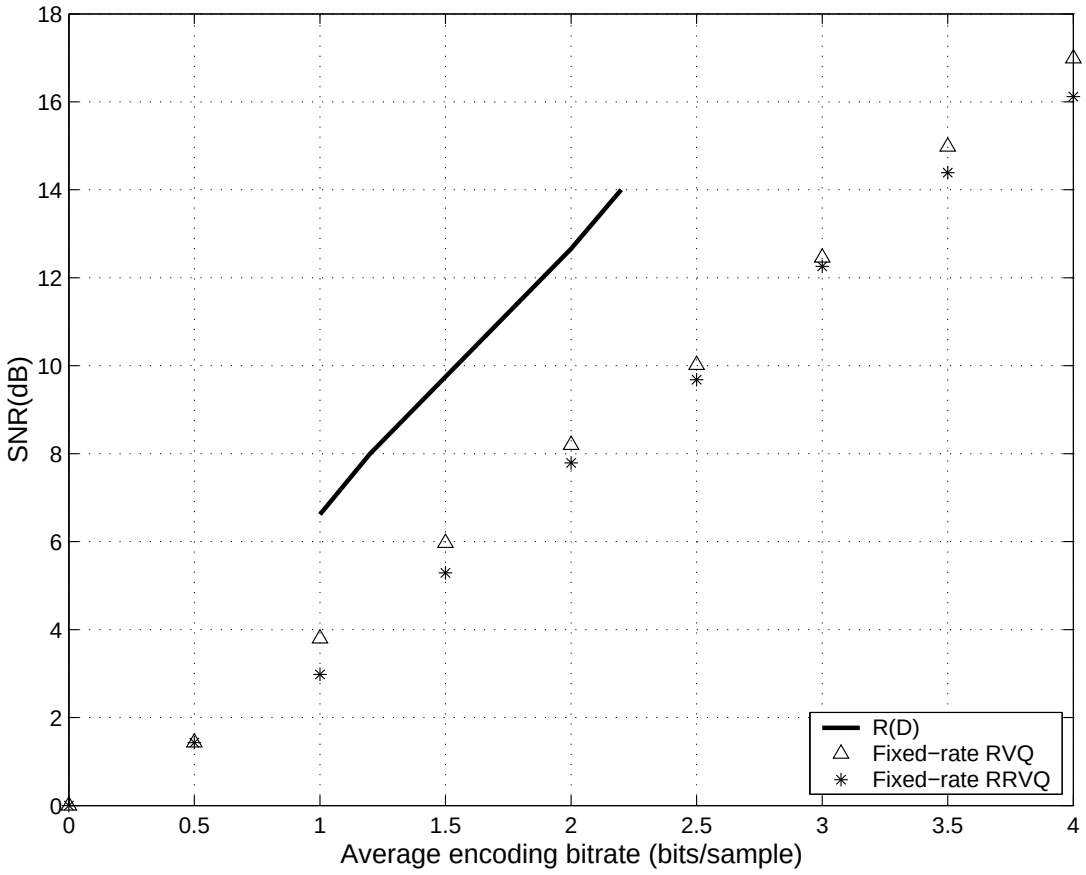


Figure 3.6: Performance comparison of RRVQ with RVQ for the memoryless Laplacian source

Chapter 4

Entropy-constrained Reflected Residual Vector Quantization

4.1 Introduction

Variable rate VQ and variable rate RVQ attracted the attention of many reaserchers in the last decade [12], [21], [24]. Some compression applications, in particular, storage and packet switched communication networks are very suited for variable rate coding [4]. With variable rate coding, some image regions can be coded with many fewer bits than other regoins depending on the amount of detail or complexity the region is. Entropy-constrained VQ (ECVQ), Entropy-constrained RVQ (EC-RVQ) are examples of variable rate coding schemes. For the RVQ, when the codewords of the direct sum codebook are allowed to be variable, the entropy for the coded input

source (image) can most often be reduced below that of the fixed rate system. Variable rate schemes usually exploit the entropy of the input signal thereby allowing a reduction in rate without a further reduction in quality [4].

One approach to constructing a variable rate VQ, in general, is simply to combine a fixed rate VQ with a variable rate noiseless code. This is done by considering the VQ codevectors to be symbols in an extend source alphabet and constructing a variable length noiseless code for these symbols. Effictively, this will assign long indexes to the least probable VQ codevector and short indexes to the most probable ones. For example, if one uses Huffman coding, the average bit rate can be reduced from the logarithm of the number of the codevectors down to the entropy of these codevectors. Therefore, a significant improvement will exist if the entropy is much less than the logarithm of the number of codevectors [4], [12], [21], [24].

Fortunately, the structure of the fixed rate RRVQ direct sum codebook (constallation) retain no diffusion between the codevectors unlike the codebook constallation of an RVQ. Moreover, since structured systems are inherently *less random* or *more ordered*, it was conjectured that in general, an RRVQ also have lower output entropy as compared to RVQ [15], [41].

4.2 The EC-RRVQ structure

For the entropy constrained design of RRVQ, the distortion trades off squared error with the codeword rate. A Lagrangian formulation is imposed, where the cost of encoding an input vector is a function of both distortion and codeword rate. The RVQ Lagrangian is defined in terms of direct sum codevectors, which is the set of all possible stage codevector sums [7], and their lengths. More specifically, we define the Lagrangian as:

$$J_\lambda = E[d(\boldsymbol{x}_1, \boldsymbol{y}(\boldsymbol{j}))] + \lambda L(\boldsymbol{j}), \tag{4.1}$$

where $\boldsymbol{x}_1$ is the input vector, $\boldsymbol{y}(\boldsymbol{j})$ is the direct sum codevector, $L(\boldsymbol{j})$ represents the length associated with the direct sum codevector $\boldsymbol{y}(\boldsymbol{j})$, and λ is the Lagrange multiplier. Consider

$$E[d(\boldsymbol{x}_p, \boldsymbol{y}_p(\boldsymbol{j}_p))] + \lambda L(j_p|j_{p-1}, j_{p-2}, \cdots, j_1) \tag{4.2}$$

to be the *pth*-stage Lagrangian where $\boldsymbol{x}_p$ is the residual vector, $\boldsymbol{y}_p(\boldsymbol{j}_p)$ is the p*th* stage codevector, and $L(j_p|j_{p-1}, j_{p-2}, \cdots, j_1)$ is the length of the codevector $\boldsymbol{y}_p(\boldsymbol{j}_p)$. In the case of EC-RVQ [12], the minimum overall Lagrangian is found by a multi-path search (*M*-search) of the stage Lagrangians. However, a fixed-rate RRVQ [10] implements a single-path search. In order to do so for an EC-RRVQ, we need to force a single-path search through the stage Lagrangians. For this purpose, we also

restrict two codevectors/stage with reflection across the mid-point plane boundary as was the case in the fixed-rate RRVQ. However, the EC-RRVQ is different from its fixed-rate counterpart in that its mid-point plane is not necessarily equidistant from codevectors. Let us elaborate on the equations for the mid-point in the case of EC-RRVQ. We assume two codevectors $\boldsymbol{y}_p(1)$ and $\boldsymbol{y}_p(2)$ per stage with associated lengths $L(1|j_{p-1}, j_{p-2}, \cdots, j_1)$ and $L(2|j_{p-1}, j_{p-2}, \cdots, j_1)$, respectively. Let the plane of equal Lagrangians at a given pth stage be defined as:

$$\begin{aligned} & \|\boldsymbol{x}_p - \boldsymbol{y}_p(1)\|^2 + \lambda L(1|j_{p-1}, j_{p-2}, \cdots, j_1) \\ = \quad & \|\boldsymbol{x}_p - \boldsymbol{y}_p(2)\|^2 + \lambda L(2|j_{p-1}, j_{p-2}, \cdots, j_1) \end{aligned} \tag{4.3}$$

Therefore, using the normal plane equation $\boldsymbol{n}_p \cdot \boldsymbol{x}_p = d$, where $\boldsymbol{n}_p$ is a normal vector associated with the plane that join $\boldsymbol{y}_p(1)$ and $\boldsymbol{y}_p(2)$, Eq. 4.3 can be rewritten as:

$$\begin{aligned} & (\boldsymbol{y}_p(1) - \boldsymbol{y}_p(2)) \cdot \boldsymbol{x}_p \\ = \quad & \frac{\|\boldsymbol{y}_p(1)\|^2 - \|\boldsymbol{y}_p(2)\|^2}{2} \\ + \quad & \frac{\lambda L(1|j_{p-1}, j_{p-2}, \cdots, j_1)}{2} \\ - \quad & \frac{\lambda L(2|j_{p-1}, j_{p-2}, \cdots, j_1)}{2} \end{aligned} \tag{4.4}$$

Then, the shortest distance from $\boldsymbol{y}_p(2)$ to the plane is given by:

$$\frac{|d| - \boldsymbol{n}_p \cdot \boldsymbol{y}_p(2)}{\|\boldsymbol{n}_p\|} \tag{4.5}$$

Hence, the mid-point for the EC-RRVQ case is:

$$\begin{aligned} \boldsymbol{m}_p = \boldsymbol{y}_p(2) + & \frac{\|\boldsymbol{y}_p(1) - \boldsymbol{y}_p(2)\|}{2}\boldsymbol{n}_p \\ + & \frac{\lambda L(1|j_{p-1}, j_{p-2}, \cdots, j_1)}{2\|\boldsymbol{y}_p(1) - \boldsymbol{y}_p(2)\|}\boldsymbol{n}_p \\ - & \frac{\lambda L(2|j_{p-1}, j_{p-2}, \cdots, j_1)}{2\|\boldsymbol{y}_p(1) - \boldsymbol{y}_p(2)\|}\boldsymbol{n}_p \end{aligned} \tag{4.6}$$

It is noticed that the mid-point is not exactly at the mid-distance but offset by an amount dependent on the difference of lengths between the two codevectors. Furthermore, the mid-point here will move in the direction of the larger length codevector. A more detailed proof is included in Appendix A.

Eq. 4.6 shows that the mid-point at a given pth residual stage depends on the codevectors as well as on the probability of the path taken from the first stage to the pth stage. Therefore, at each stage 2^{p-1} mid-points are required to be calculated, where $1 < p < P$. For example, for the third stage we need to calculate $2^2 = 4$ mid-points.

4.2.1 Operation of the EC-RRVQ Algorithm

Generally, the plane of equal Lagrangians will fall between the two codevectors. However, when two codevectors have very different lengths, the resulting difference may move the mid-point or equivalently the plane on the far side of one of the codevectors. For this case, we are not able to apply reflection symmetry. For a given pth stage, there are 2^{p-1} mid-points. Some of them lie in between the two codevectors, the others may not. Thus we have a mix of reflected and unreflected residuals for the $(p+1)$th residual stage. To avoid this mixing, we have adopted a convention of announcing a given pth stage reflected if all the mid-points lie in between the codevectors. Otherwise, if any of them fails to do so, we announce the stage unreflected and avoid performing reflection in that stage. Therefore, generally speaking, an EC-RRVQ encoder will consist of some reflected and some unreflected residual stages.

In order to find a set of points on the convex hull of the operational rate-distortion curve, the minimization of J_λ is repeated for various $\lambda's$. The algorithm starts by designing a fixed codeword length RRVQ, i.e, when $\lambda = 0$. Then, using a predetermined sequence of $\lambda's$ computed as reported in [13], the algorithm provides a set of locally optimal EC-RRVQ codebooks with various bit rates. More specifically, given a training set of vectors and a fixed λ_i, the *ith* EC-RRVQ codebook is designed by iterating the following steps:

1. For each residual stage, check the mid-points calculated from various tree paths to see if all of them lie on the line joining two codevectors. If yes, then announce that stage as a reflected stage and obtain the residuals after reflection. However, if the answer is no, then announce that stage as an unreflected stage, and obtain the residuals without performing any reflection.

2. Encode the training set vectors with an *old* reflected direct sum codebook which minimizes J_{λ_i}.

3. Map the fixed-length indices of the direct sum codevectors into variable-length indices so that the average output rate is reduced.

4. Replace the *old* stage codevectors with *new* codevectors that are centroids of their reflected residuals. Whenever the relative reduction in average distortion is smaller than a predefined threshold, stop the iterations. Fig. 4.1 demonstrates this EC-RRVQ algorithm.

The preceding EC-RRVQ algorithm described above draws some similarity with the entropy-constrained scalar quantizer (EC-SQ) algorithm reported in [16]. However, in [16], the algorithm is considered as unstable. In fact, the mid-points (threshold levels in [16]) may not converge at all. In addition, for multiple fixed points, the algorithm is unable to find all the fixed points. In this work, we do not apply reflection for each stage, rather, we implement an EC-RRVQ with a mix of reflected as well as unreflected residual stages. Based on the empirical experimental results,

we will demonstrate that this mix, generally, stabilizes the optimization algorithm and is convergent. However, the monotonic convergence properties are lost. Nevertheless, in the absence of a better strategy to design and implement EC-RRVQ codes, an ad hoc scheme is adopted as explained in the coming sections.

4.2.2 Complexity of the EC-RRVQ Algorithm

The RRVQ has only two codevectors and a single dot product calculation for deciding the best stage codevector. Hence, the computational cost of an EC-RRVQ will be given by $C_{EC-RRVQ} = P + O(R)$ vector distortion calculations per source vector, where $O(R)$ represents the average number of reflections required to encode the source vectors and P represents the number of stages. On average, reflection occurs at one half of the RRVQ stages [15], i.e, $O(R) = \frac{1}{2}P$. For example, considering 16 residual stages, the EC-RRVQ will require only 24 vector distortion calculations per source vector. On the other hand, for an entropy-constrained M-search residual quantizer (EC-RVQ) encoder, the computational cost is given by:

$$C_{EC-RVQ} = N_1 + \sum_{p=1}^{P} [min(M, \prod_{\rho}^{p} N_p) \times N_p] \tag{4.7}$$

vector distortion calculations per source vector, where N_1 is the size of the first-stage codebook, N_p is the size of the *pth*-stage codebook and M is the number of paths searched [10]. For a 2-codevector/stage (binary) EC-RVQ with $M = 4$ (four saved

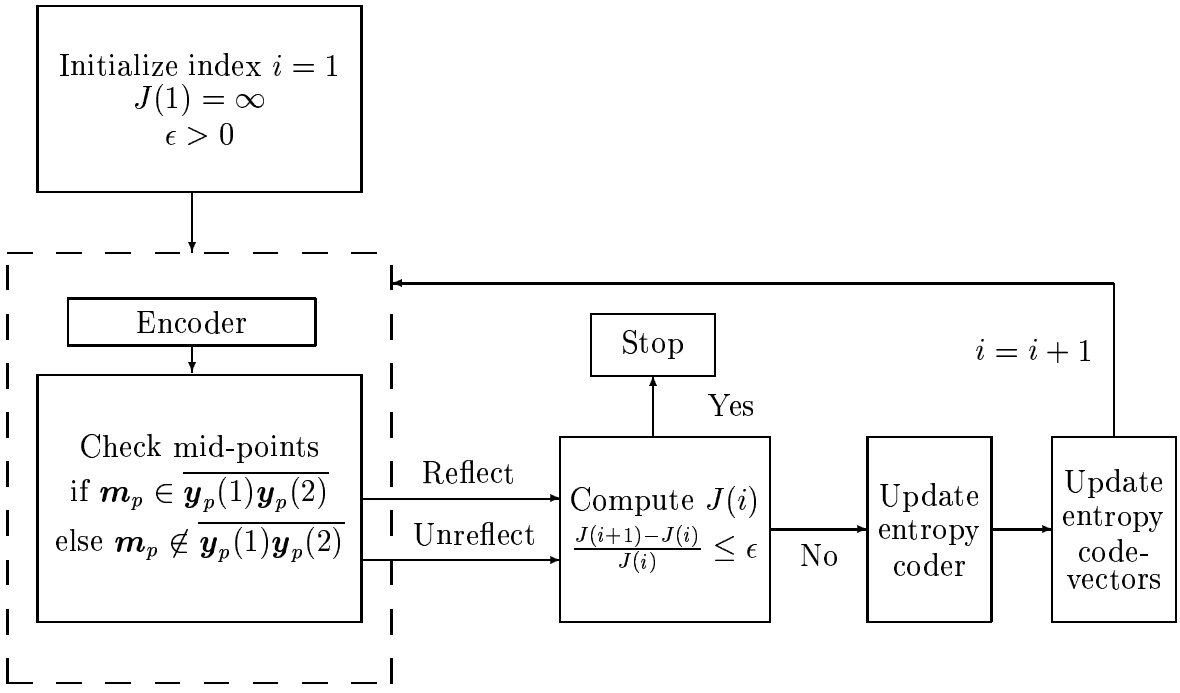


Figure 4.1: The EC-RRVQ design algorithm

paths at each stage), the encoder complexity will be:

$$C_{EC-RVQ} = \begin{cases} 2 & \text{if } P = 1 \\ 6 + 8(P-2) & \text{if } P \geq 2 \end{cases} \tag{4.8}$$

For example, for a 16-stage EC-RVQ, the encoder will require 118 vector distortion calculations per source vector. Fig. 4.2 shows a comparison of encoding complexity between EC-RVQ and EC-RRVQ for an increasing bit rate. At very low bit rates, say $P \leq 3$, the complexity for both encoders is small. However, as P increases, the gap between the curves becomes larger and larger indicating that EC-RRVQ is working at nearly a *constant* function of bit rate.

An important complexity-reducing feature of EC-RRVQ is its potential to use *stage-conditional* entropy tables of relative sizes, where conditioning is performed on previous stages [41]. In other words, the lengths of the pth stage codevectors are approximated by making a Markov-like assumption and using conditional probabilities that depend only on the last $m < p$ stages. With the use of a smaller Markov model order m, a large reduction in entropy-tables storage can be obtained. The length for a direct sum codevector is given by:

$$L(\boldsymbol{j}) = L(j_1) + L(j_2|j_1) + \cdots + L(j_P|j_{P-1}, j_{P-2}, \cdots, j_1) \tag{4.9}$$

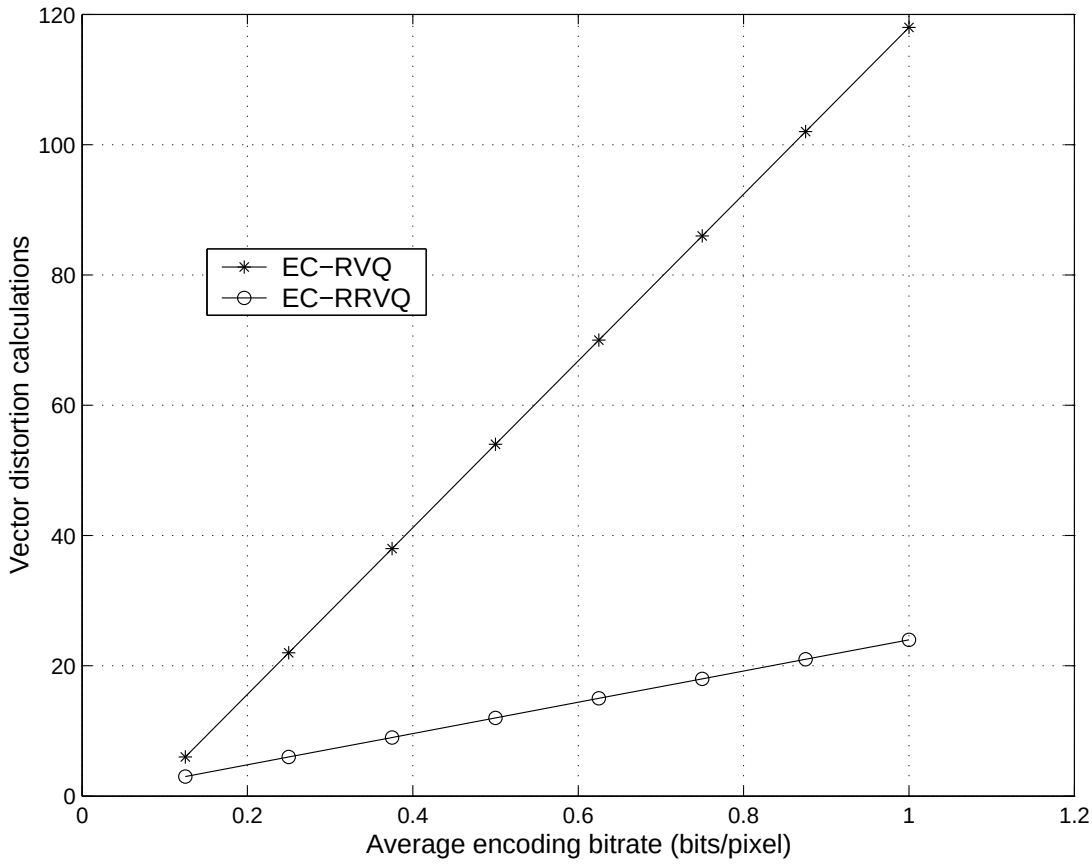


Figure 4.2: Encoding complexity of EC-RVQ and EC-RRVQ

For a given Markov model order m Eq. 4.9 can be approximated as:

$$L(\boldsymbol{j}) = L(j_1) + L(j_2|j_1) + \cdots + L(j_P|j_{P-1}, j_{P-2}, \cdots, j_{P-m}) \tag{4.10}$$

where $P - m >> 0$. The experiments reported in this thesis utilize Markov model order as small as two and one.

4.3 EC-RRVQ Performance

4.3.1 Synthetic Sources

Simulations were performed for zero-mean unit variance Gaussian and Laplacian data. Training data were made of 2,000,000 samples. The resulting codebooks were tested for 100,000 samples outside the training set. Both EC-RVQ and EC-RRVQ were simulated for a two-dimensional vector source with full Markov order. Initially, the peak bit rate of both quantizers is fixed to 4 bits/sample. Comparisons were made between a 4-stage EC-RVQ with M=7 and stage codebook size 4, and an 8-stage EC-RRVQ. Fig. 4.3 and Fig. 4.4 indicate that EC-RRVQ outperforms RRVQ, RVQ, and EC-RVQ for both Gaussian and Laplacian sources, respectively. The difference in SNR between EC-RRVQ and RRVQ was 4-dB on average. In addition, RVQ exhibited a better performance than RRVQ but, at the same time, EC-RRVQ has higher performance than EC-RVQ. This is due to the fact that RRVQ

has added constraint of reflection on its direct sum codebook as compared to the RVQ direct sum codebook. Consequently, this constraint provided more symmetry and order in the direct sum codebook formation which resulted in higher distortion but lower output entropy. Therefore, an improved performance for the EC-RRVQ was obtained.

4.3.2 Natural Images

Experimental results are used to study the effect of joint decoder optimization, Markov model order, and *peak* bit rate on the performance of the EC-RRVQ. In addition, comparisons are made of the performance of the EC-RRVQ with those of the EC-RVQ.

The training set consists of eight 512×512 monochrome images taken from the USC database. Shifts and rotations are used to generate additional training vectors, leading to more than $250,000$ of 4×4 vectors. The test images used in the experiments are the well-known LENA (cropped part from the original coded LENA image is shown) and BOAT images. Both images are excluded from the training set.

Joint Decoder Optimization

For adopting a jointly optimized decoder, the old stage codevectors are iteratively replaced by the stage-removed residual centroids of their respective sets (i.e. using

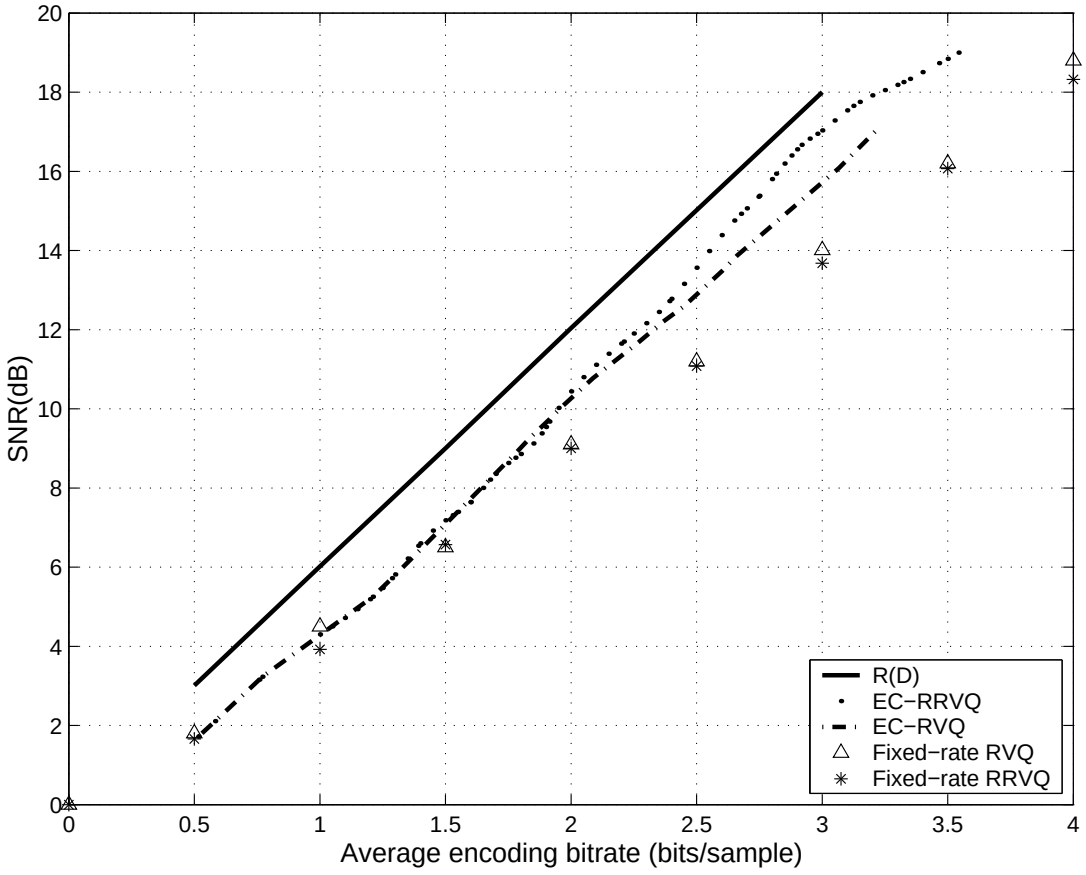


Figure 4.3: Performance comparison of EC-RRVQ with various source coding schemes for the memoryless Gaussian source

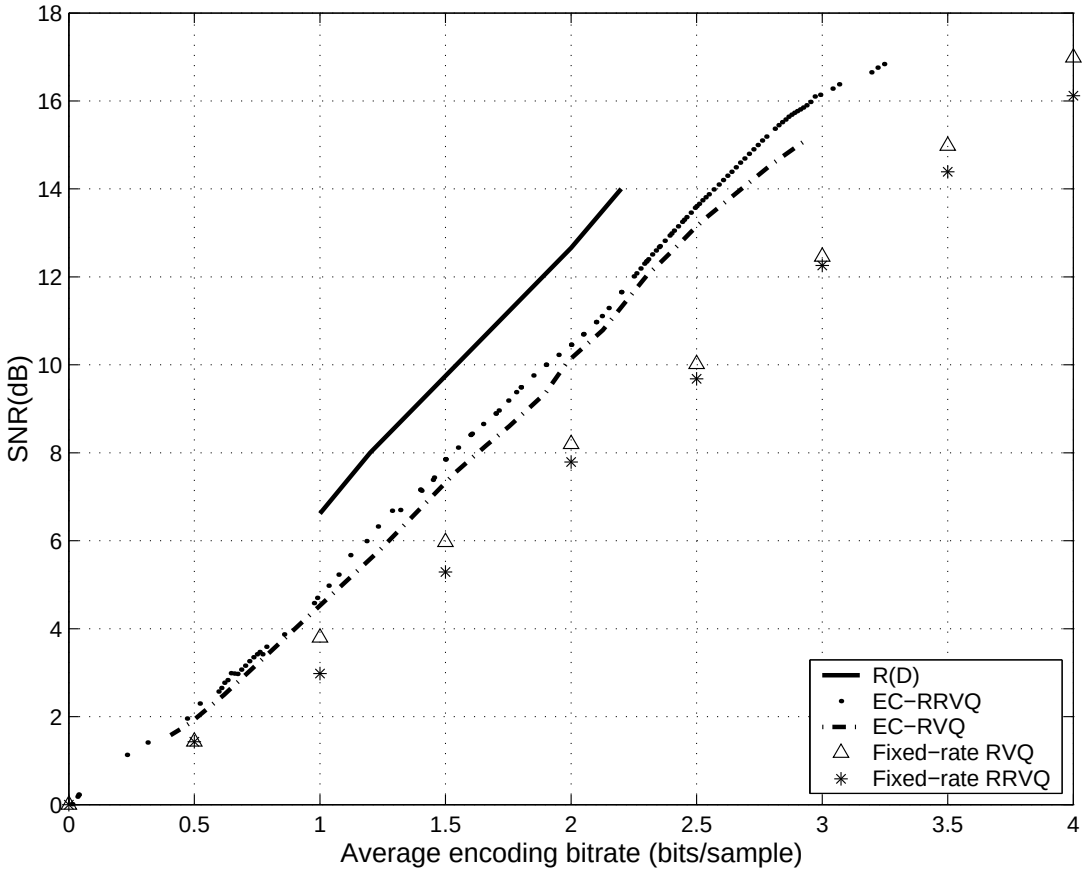


Figure 4.4: Performance comparison of EC-RRVQ with various source coding schemes for the memoryless Laplacian source

the Gauss-Seidel iterative procedure). On the other hand, a non-joint decoder optimization, which is known as the stage-sequential decoder optimization, involves only one iteration of the Gauss-Seidel algorithm [8],[24]. Here, we want to compare the PSNR performance by utilizing both the joint and non-joint decoder optimization.

Fig. 4.5 represents the performance of an EC-RRVQ designed with both jointly and non-jointly optimized decoders as done in [24]. The initial RRVQ codebook consists of 16 stage codebooks. Hence, the initial RRVQ codebook size corresponds to a peak bit rate of 1.00 bits/pixel (bpp). The Markov model order is one ($m = 1$). Based on that experiment, there exits a difference of 0.5 dB on average for rates higher than 0.5 bpp. However, as the bit rate goes below 0.5 bpp, the gap between the two curves starts to decrease. At rates less than 0.45 bpp, the two curves start to coincide. In consequence, at low bit rates, there seems to be no advantage in employing a joint-decoder optimization.

Markov Model Order

It is of particular interest to find the performance gain as a function of the Markov model order m. The experiment will assist us in determining how large m should be to obtain satisfactory performance. Fig. 4.6 shows the result of the experiment. The initial RRVQ codebook employed consists of 16 stages resulting in a peak bit rate of 1.00 bpp. Three curves are shown for Markov model order one, two and three, respectively. We can see that the PSNR performance of the EC-RRVQ with

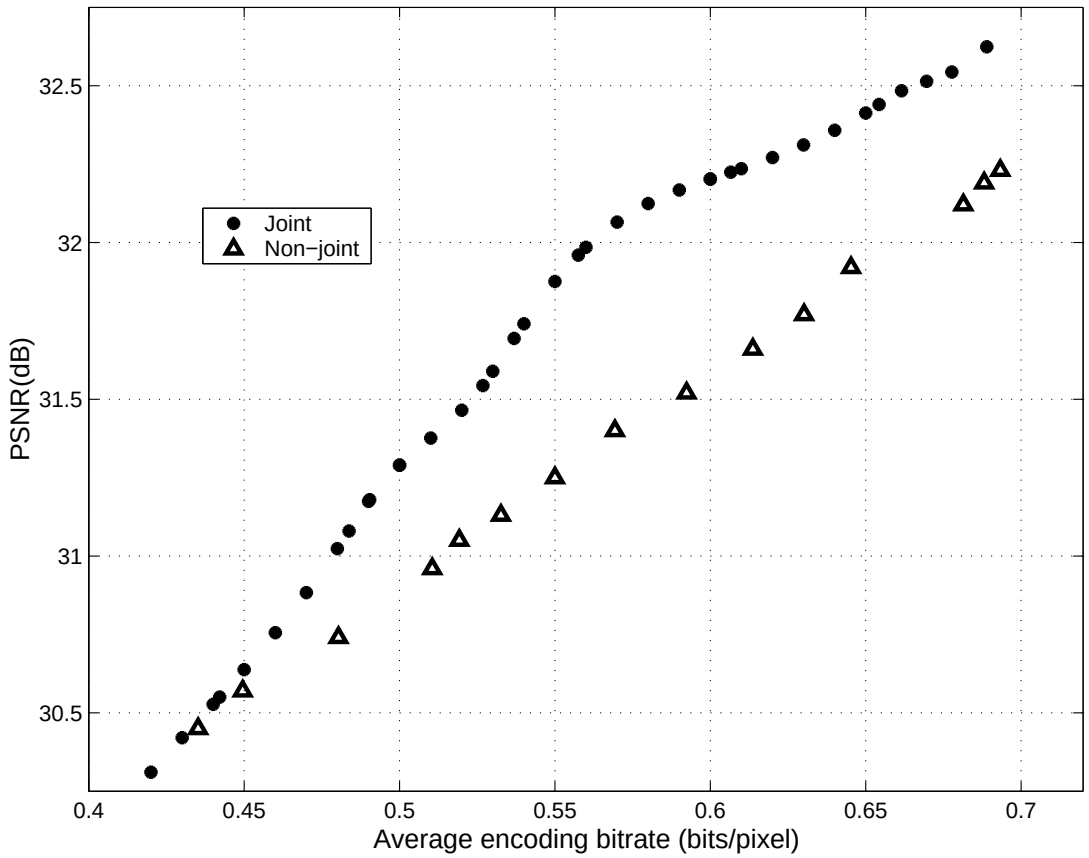


Figure 4.5: PSNR performance for the test image LENA using both jointly and non-jointly optimized decoders (The number of stages is 16 and $m = 1$)

Markov model order one is nearly identical to that of order two. This shows that an EC-RRVQ encoder has the potential to get a nearly optimal entropy coding performance by employing a small Markov model order such as one or at most two. In addition, an advantage of using small values for m is the great reduction in memory requirement.

Peak Bit Rate

In this part, two EC-RRVQ's were designed and used to encode the test image LENA. The first one was initialized with a 16-stage RRVQ codebook corresponding to a peak bit rate of 1.00 bpp. The second EC-RRVQ was initialized with a 28-stage RRVQ codebook giving 1.75 bpp as a peak bit rate. In both cases, the Markov model was of the first order ($m = 1$). Fig. 4.7 shows the performance of both EC-RRVQ's with their own specifications as mentioned earlier. As expected, the EC-RRVQ with large peak bit rate outperforms the one with a smaller peak bit rate.

Unreflected Stages

As discussed before, during the design of the EC-RRVQ codebook, whenever the mid-point lies between the stage codevectors, announce the stage as a reflected stage and obtain the residuals after performing reflection. Otherwise, the stage is announced as an unreflected stage and the residuals are obtained without performing any reflection. Therefore, a given EC-RRVQ encoder consists of a mix of reflected

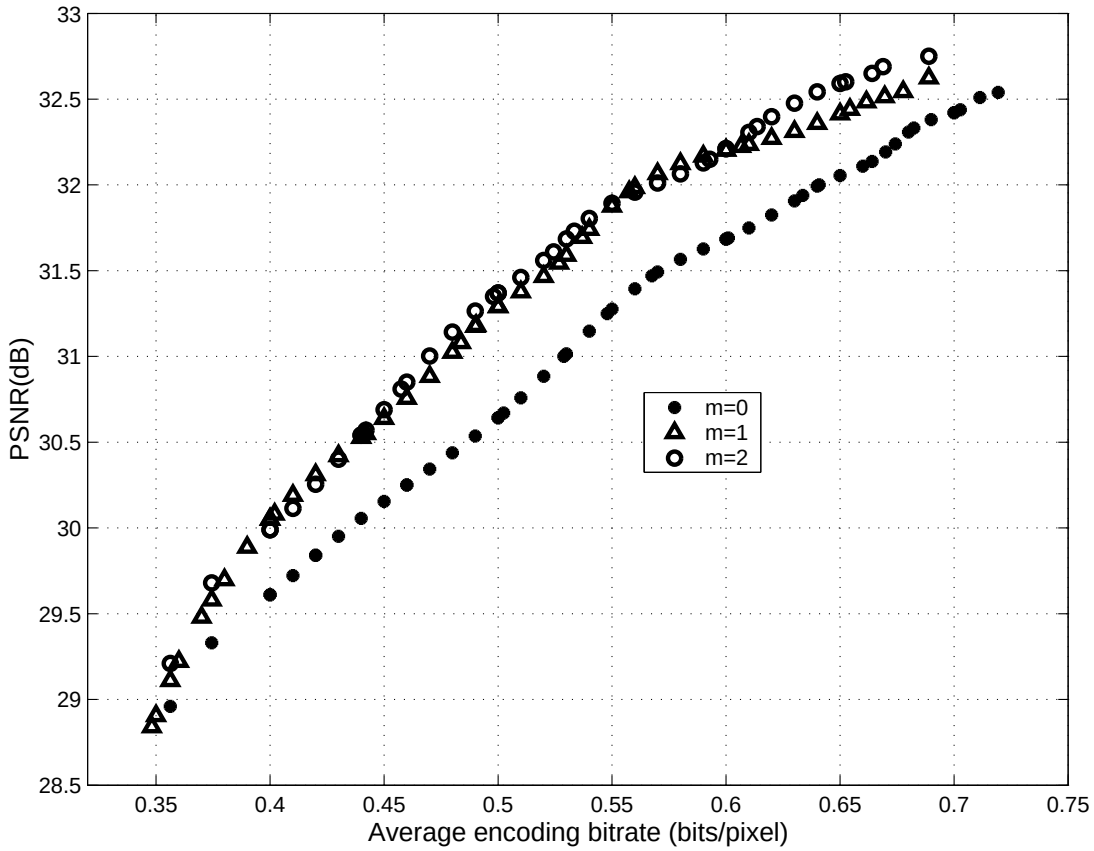


Figure 4.6: Rate-distortion performance of EC-RRVQ with 16 stages for the test image LENA at increasing values of m (The vector size is 4×4)

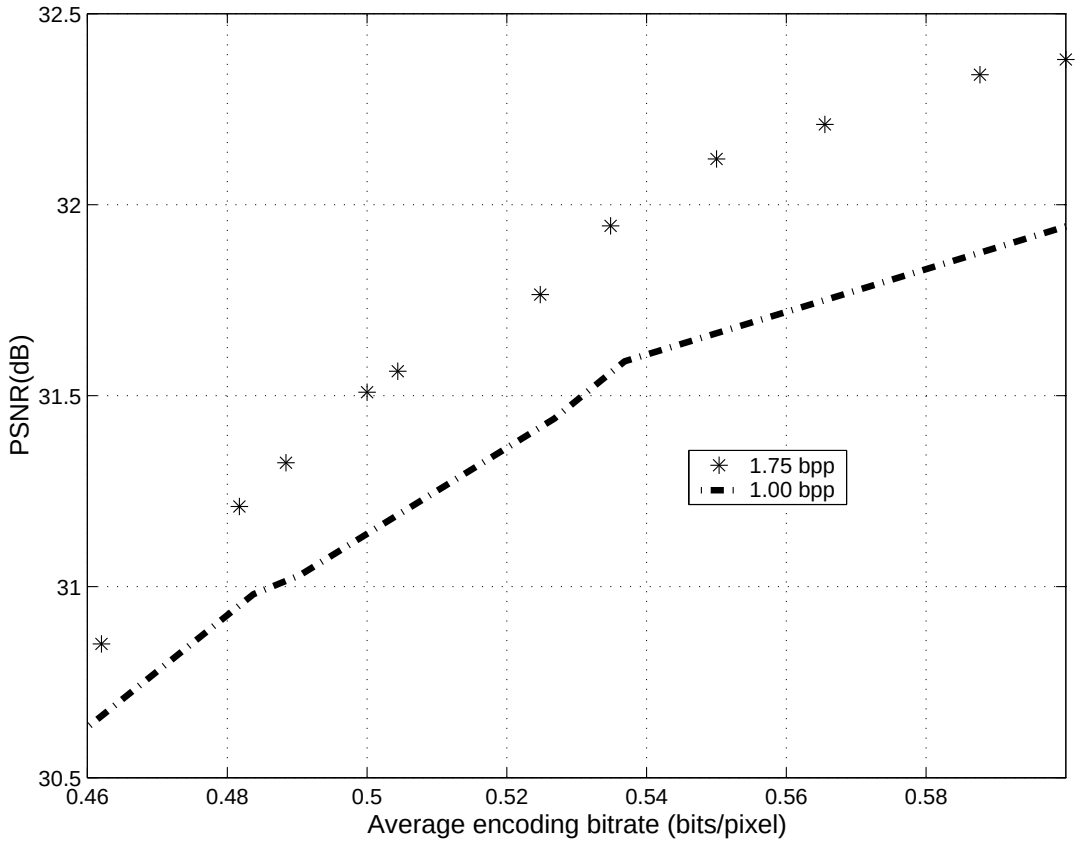


Figure 4.7: Rate-distortion performance of EC-RRVQ for the test image LENA at two different peak bit rates. The top one is for 28 stages giving 1.75 bpp and the bottom curve is for 16 stages giving 1.00 bpp (The vector size is 4×4 and $m = 1$)

and unreflected stages. In this section, we provide plots that describe the effect of Markov model order m and bit rate on the mixing. A 16-stage EC-RRVQ encoder was tested for the purpose of finding how many out of 16 would become unreflected stages. Fig. 4.8 shows such an experiment. For $m = 0$, at rates 0.633 and 0.528 bpp, we obtianed one unreflected stage. For $m = 1$, at 0.490 bpp, one stage was unreflected while at rate 0.37 bpp, two stages were announced as unreflected stages. This shows that as the Markov model order m increases, there is a greater chance of getting a higher number of unreflected stages for a given bit rate.

Comparsions with EC-RVQ

Fig. 4.9 compares the distortion versus rate performance on LENA for both EC-RRVQ and binary EC-RVQ. Initially, the codebooks for both the RRVQ and RVQ were designed using 16 stages with $m = 1$. For the EC-RVQ, the number of saved paths (M) was equal to four. Looking at the EC-RRVQ and EC-RVQ performances, we notice that EC-RRVQ provides at least 1.5 dB improvement over the EC-RVQ at rates higher than 0.7 bpp. Between 0.5 and 0.7 bpp, EC-RRVQ is outperforming EC-RVQ by 1.0 dB on the average. The gap between two entropy-constrained designs reduces with the reduction in bit rates and the two curves superimpose each other below 0.37 bpp.

Finally, the LENA image (a cropped region is shown) and the BOAT image were both used for subjective quality evaluation. Fig. 4.10 (a) shows the original LENA

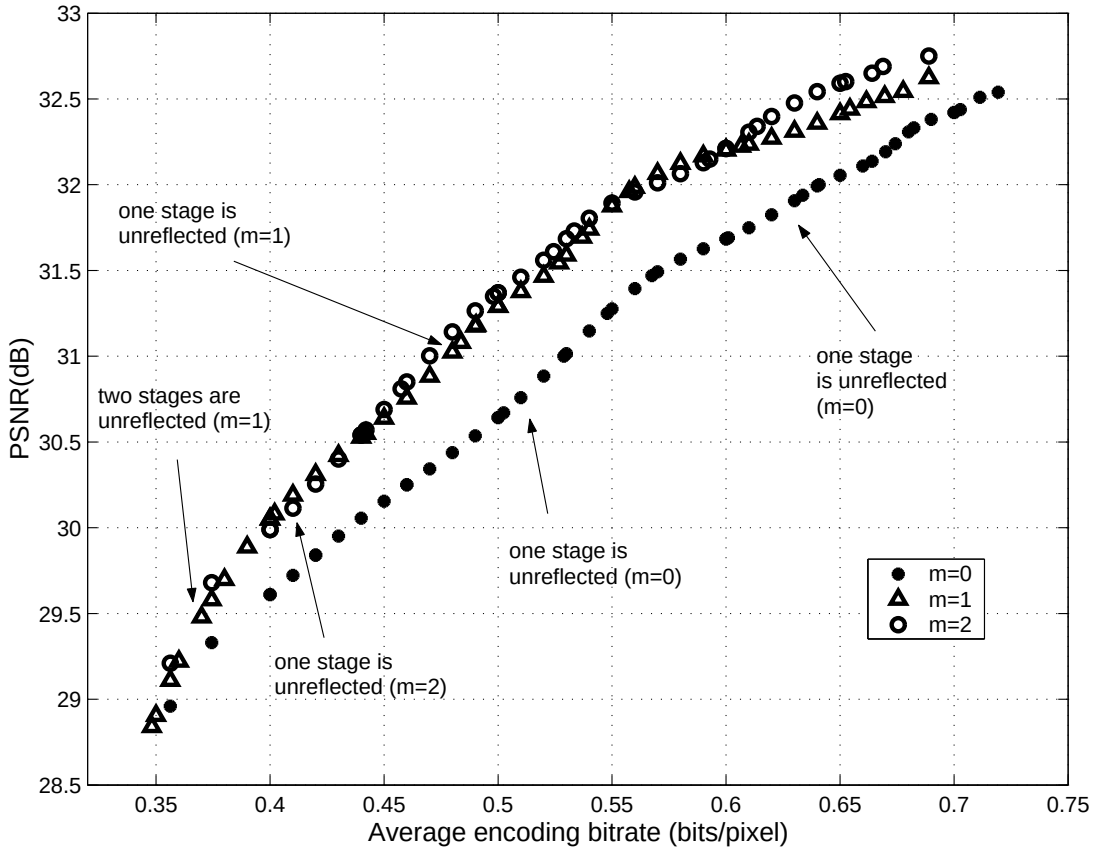


Figure 4.8: Rate-distortion performance of EC-RRVQ showing the rates at which stages were unreflected for the test image LENA . The number of stages was 16 at an increasing values of m (The vector size is 4×4)

image, Fig. 4.10 (b) the reconstructed LENA image using EC-RVQ at a rate of 0.527 bpp and Fig. 4.10 (c) the reconstructed LENA image using EC-RRVQ at a rate of 0.526 bpp. The EC-RRVQ coded image has a better PSNR as compared to that of the EC-RVQ coded image. The difference in PSNR is about 1 dB. Similarly, the BOAT image at Fig. 4.11 (a) was coded using EC-RVQ at a rate of 0.641 bpp displayed in Fig. 4.11 (b) while Fig. 4.11 (c) shows the reconstructed BOAT image using EC-RRVQ at a rate of 0.593 bpp. The difference in PSNR was 0.4 dB. Generally speaking, the difference manifests itself in the form of smaller blocking artifacts, less false contouring, and higher contrast for the EC-RRVQ coded images.

Table 4.1 shows PSNR comparisons of EC-RRVQ and EC-RVQ for three test images at an output bit rate of roughly 0.5 bpp. The table indicates that EC-RRVQ outperforms EC-RVQ in PSNR performance while demanding a lower encoding complexity.

Table 4.1: PSNR OF EC-RVQ AND EC-RRVQ FOR FOUR TEST IMAGES TAKEN FROM THE USC DATABASE (THE BIT RATE IS 0.5 BPP AND THE VECTOR SIZE IS 4×4)

	EC-RVQ,peak=1.00 bpp	EC-RRVQ,peak=1.00 bpp
LENA	30.61	31.41
BOAT	28.2	28.41
BRIDGE	25.11	25.2

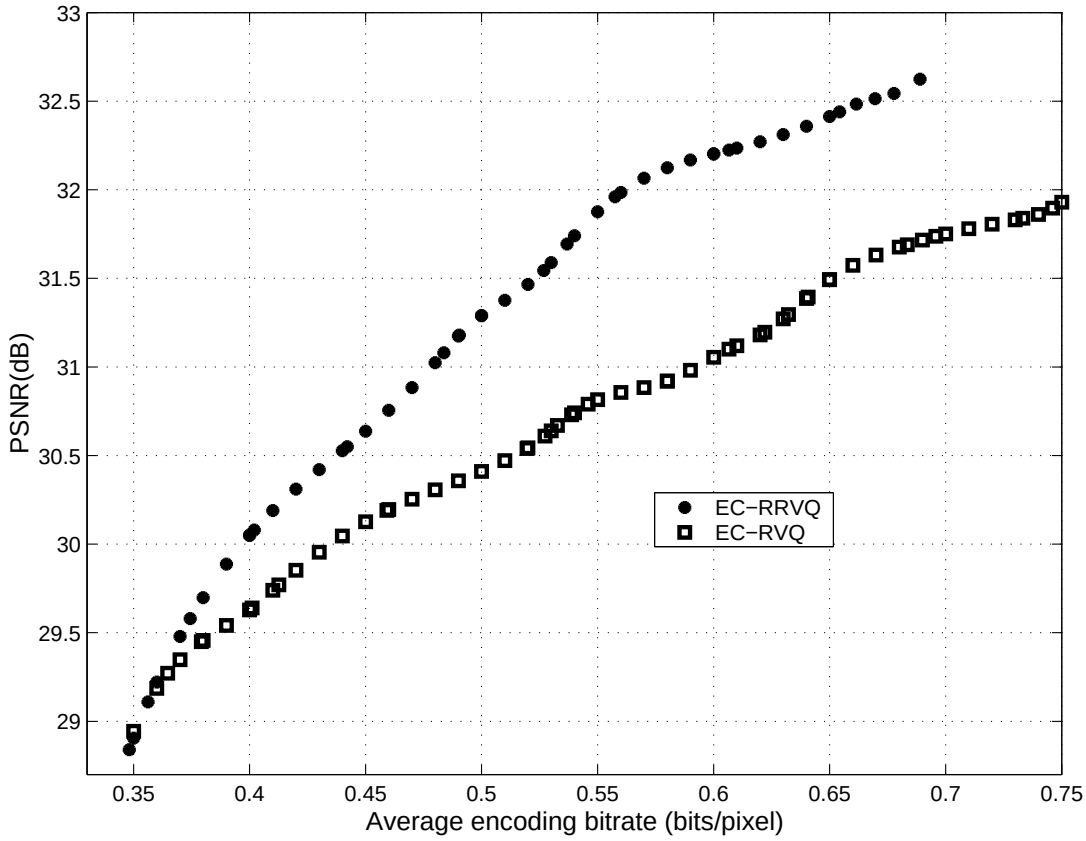


Figure 4.9: Rate-distortion performance of EC-RRVQ and EC-RVQ with 16 stages for the test image LENA at $m = 1$ (The vector size is 4×4)

(a)

(b)

(c)

Figure 4.10: Cropped Image LENA coded using (b) EC-RVQ at a bit rate of 0.527 bpp with PSNR of 30.61 dB (c) EC-RRVQ at a bit rate of 0.526 bpp with PSNR of 31.41 dB

(a)

(b)

(c)

Figure 4.11: Image BOAT coded using (b) EC-RVQ at a bit rate of 0.641 bpp with PSNR of 28.82 dB (c) EC-RRVQ at a bit rate of 0.593 bpp with PSNR of 29.2 dB

4.3.3 Extension to large block EC-RRVQ

Direct use of VQ on image pixels suffers from a serious complexity barrier and is limited to rather modest vector dimensions and codebook sizes. However, there are various applications that require large vector sizes including variable dimension VQ and transform VQ [20], [4]. Multispectral imagery and video coding applications also benefit from the use of large vector sizes.

Practical entropy-constrained vector quantizers (EC-VQ)'s are limited to sizes of 4×4 [13]. Also, experimental results for both EC-VQ and an entropy-pruned TSVQ with 8×8 do not exist [24]. According to [42], [15], the performance degradation of an RVQ quantizer, in general, is due to two factors. The first factor is that the decoder is constrained to have a direct sum codebook structure. The other factor is the existence of entanglements in the RVQ tree that will complicate the task of optimizing stagewise partitions. For the RRVQ, a binary RVQ is used with an additional to the reflection constraint. A binary RVQ is most efficient in terms of memory. The reflection constraint will avoid RVQ tree entanglements and make the single path search optimal [15]. Finally, by minimizing the output entropy of the RRVQ, the EC-RRVQ achieved a better quantizer performance in addition to the tremendous encoder complexity savings compared to the EC-RVQ. We believe that the EC-RRVQ, due to its smaller encoding complexity, presents itself as a practically feasible and better large-dimensional VQ encoder than the EC-RVQ.

In [42], authors used the M-search algorithm for the RVQ and EC-RVQ for block dimension of 16×16. The number of vectors per stage was equal to 16 with 16 multi-paths. The results were reasonable. However, the main disadvantage is that the encoder computations are also increased by a factor of 16. Since the EC-RRVQ encoder complexity depends only on the number of stages rather than multi-paths, it is worth testing the EC-RRVQ for a large block RRVQ with dimensions of 8×8 and 16×16.

Blocks of size 8×8

The training set for an 8×8 vector dimension contained no more than $500,000$ vectors and 32 fixed rate RRVQ stages were designed giving 0.5 bpp as a peak bit rate. The experiments were performed for obtaining satisfactory performance as a function of Markov model order. Fig. 4.12 shows that for rates between 0.25 and 0.4 bpp, there exists a difference of approximately 1 dB on average between EC-RRVQ with $m = 0$ and $m = 1$. Also, for the same range of rates, the difference between EC-RRVQ with $m = 1$ and $m = 2$ is roughly $0 - 0.2$ dB, which indicates that first order Markov model is providing a satisfactory rate-distortion performance. In contrast, for rates less than 0.25 bpp, all curves provide a similar performance. Therefore, for low bit rates there is no advantage in using Markov orders higher than 0. Fig. 4.13 shows the LENA image coded at a bit rate of (a) 0.349 bpp with PSNR of 30.49 dB (b) 0.257 bpp with PSNR of 28.99 dB (c) 0.177 bpp with PSNR of 28.15 dB and

(d) 0.129 bpp with PSNR of 26.29 dB.

A comparison in Fig. 4.14 is made between EC-RRVQ and EC-RVQ for the same set of data and dimension of 8×8 for $m = 1$. For rates higher than 0.3 bpp, the EC-RRVQ outperforms the EC-RVQ by an amount of 1 dB. However, for rates less than 0.3 bpp, the two curves become almost identical. For a subjective comparison, Fig. 4.15 is provided. It shows the test image LENA coded using (a) EC-RVQ at a bit rate of 0.179 bpp with PSNR of 28.03 dB (b) EC-RRVQ at a bit rate of 0.177 bpp with PSNR of 28.15 dB both of vector dimension 8×8 and $m = 1$.

Blocks of size 16×16

For the 16×16 vector dimension, the training set contained more than $350,000$ vectors. The same work as in the previous subsection was done with vectors of size 16×16. For the Markov model order experiment, the peak bit rate was 0.25 bpp, meaning 64 fixed rate stages were designed. Fig. 4.16 shows that for rates between 0.1 and 0.2 bpp, there exists a difference of approximately 0.5 dB on average between EC-RRVQ with $m = 0$ and $m = 1$. Also, for the range of rates between 0.18 and 0.2, the difference between EC-RRVQ with $m = 1$ and $m = 2$ is 0.2 dB. In contrast, for rates less than 0.1 bpp, all curves provide similar performance. Fig. 4.17 shows the LENA image coded at a bit rate of (a) 0.200 bpp with PSNR of 29 dB (b) 0.164 bpp with PSNR of 27.83 dB (c) 0.106 bpp with PSNR of 26.11 dB (d) 0.066 bpp

with PSNR of 24.76 dB.

A comparison is made between EC-RRVQ and EC-RVQ for the same set of data and vector dimension of 16×16 for $m = 1$. It can be seen from Fig. 4.18 that for rates higher than 0.12 bpp, the gap between the EC-RRVQ and the EC-RVQ curves is about 0.6 dB on average, while for rates less than 0.12 bpp, the two curves will join. For a subjective comparison, Fig. 4.19 is provided. It shows the test image LENA coded using (a) EC-RVQ at a bit rate of 0.215 bpp with PSNR of 28.39 dB (b) EC-RRVQ at a bit rate of 0.201 bpp with PSNR of 29 dB both of dimension 16×16 and $m = 1$.

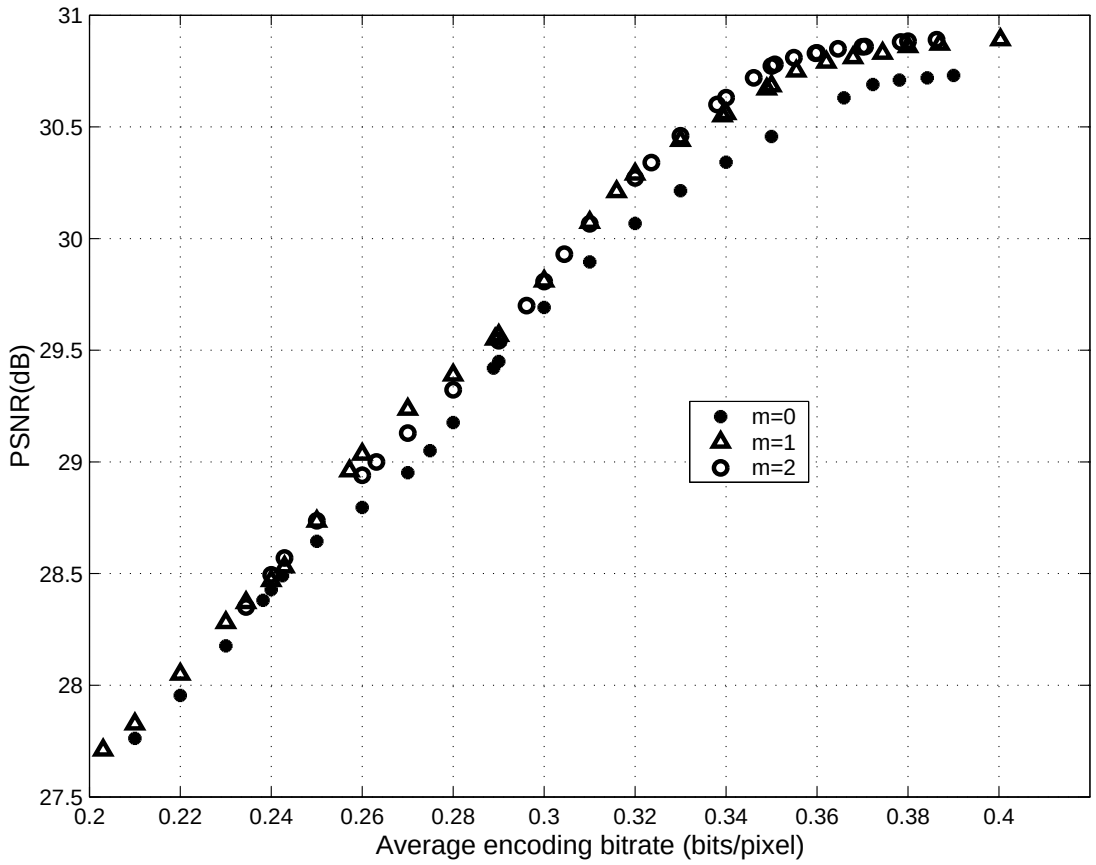
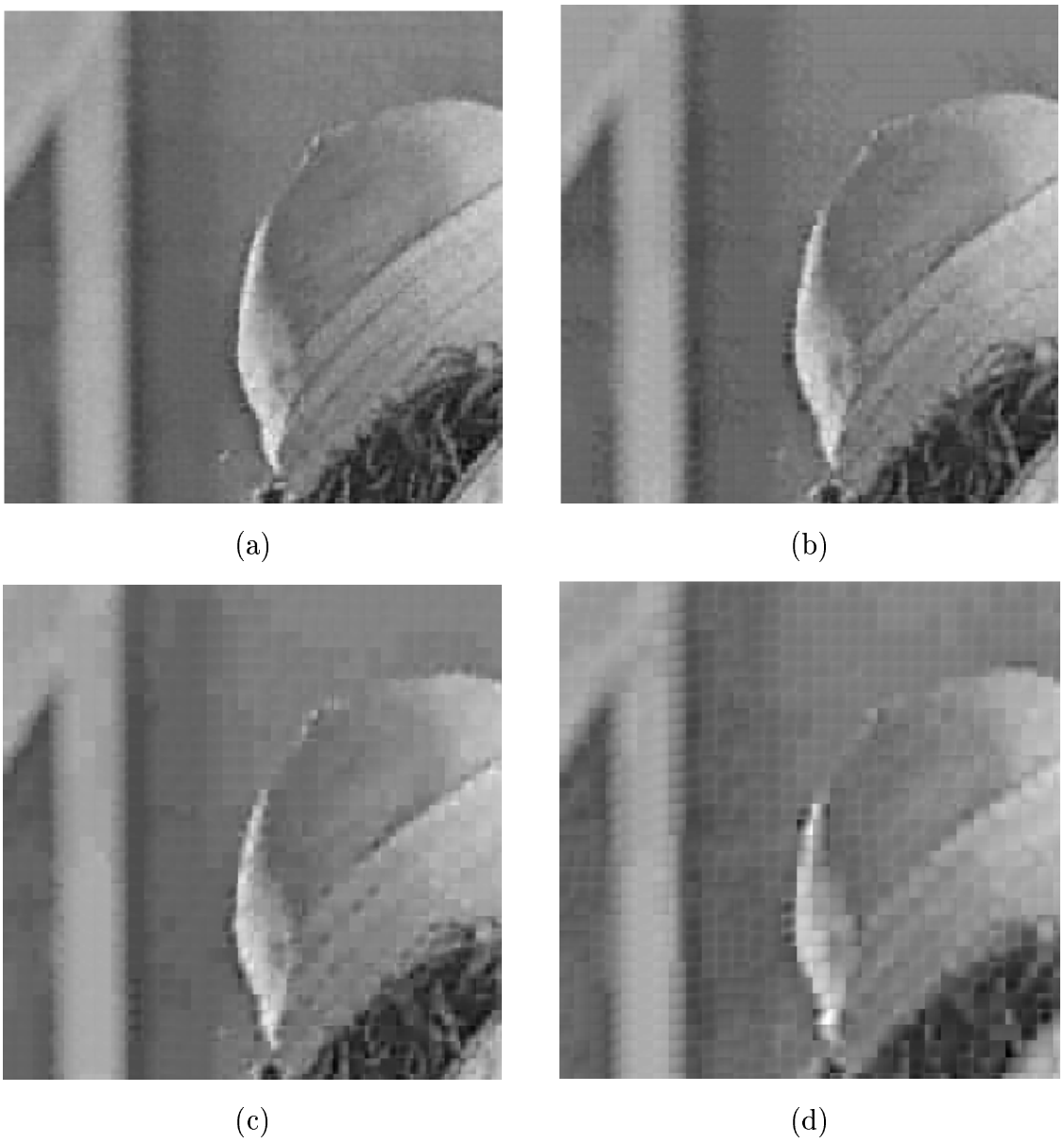


Figure 4.12: Rate-distortion performance of EC-RRVQ with 32 stages for the test image LENA at increasing values of m (The vector size is 8×8)



(a) (b)

(c) (d)

Figure 4.13: Cropped Image LENA coded using EC-RRVQ of dim=8×8 and $m = 1$ at a bit rate of (a) 0.349 bpp with PSNR of 30.49 dB (b) 0.257 bpp with PSNR of 28.99 dB (c) 0.177 bpp with PSNR of 28.15 dB (d) 0.129 bpp with PSNR of 26.29 dB

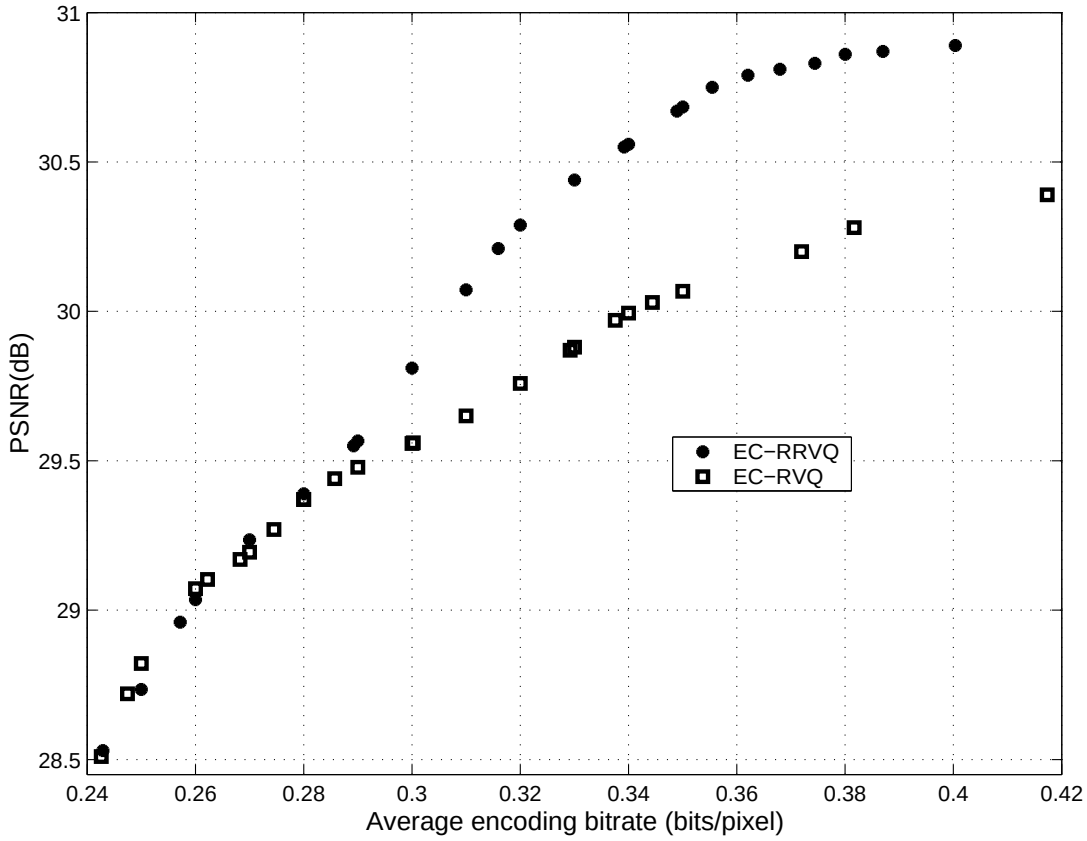
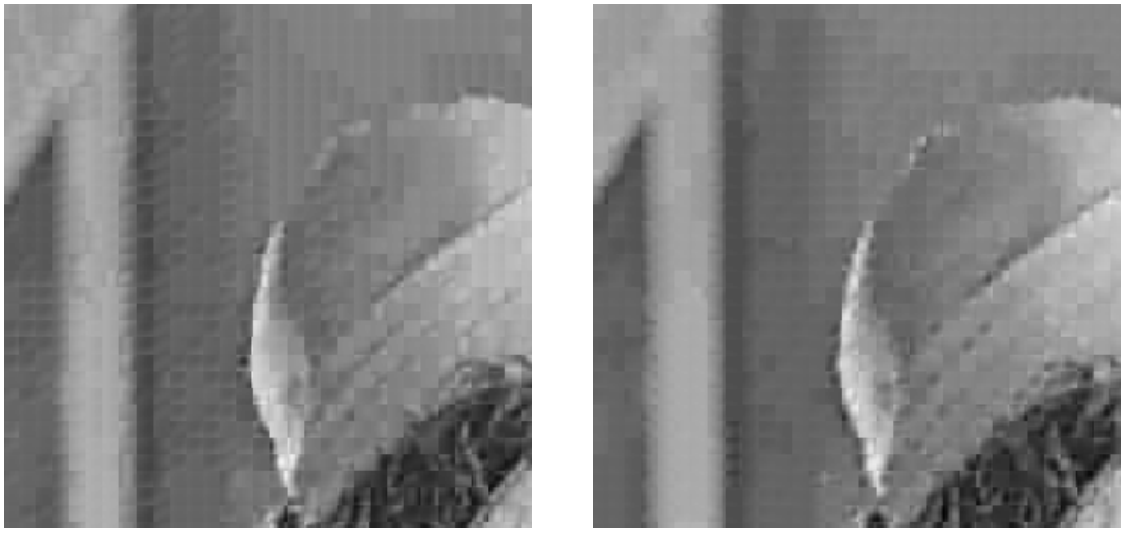


Figure 4.14: Rate-distortion performance of EC-RRVQ and EC-RVQ with 32 stages for the test image LENA at $m = 1$ (The vector size is 8×8)



(a) (b)

Figure 4.15: Cropped Image LENA coded using (a) EC-RVQ at a bit rate of 0.179 bpp with PSNR of 28.03 dB (b) EC-RRVQ at a bit rate of 0.177 bpp with PSNR of 28.15 dB both of dimension 8×8 and $m = 1$

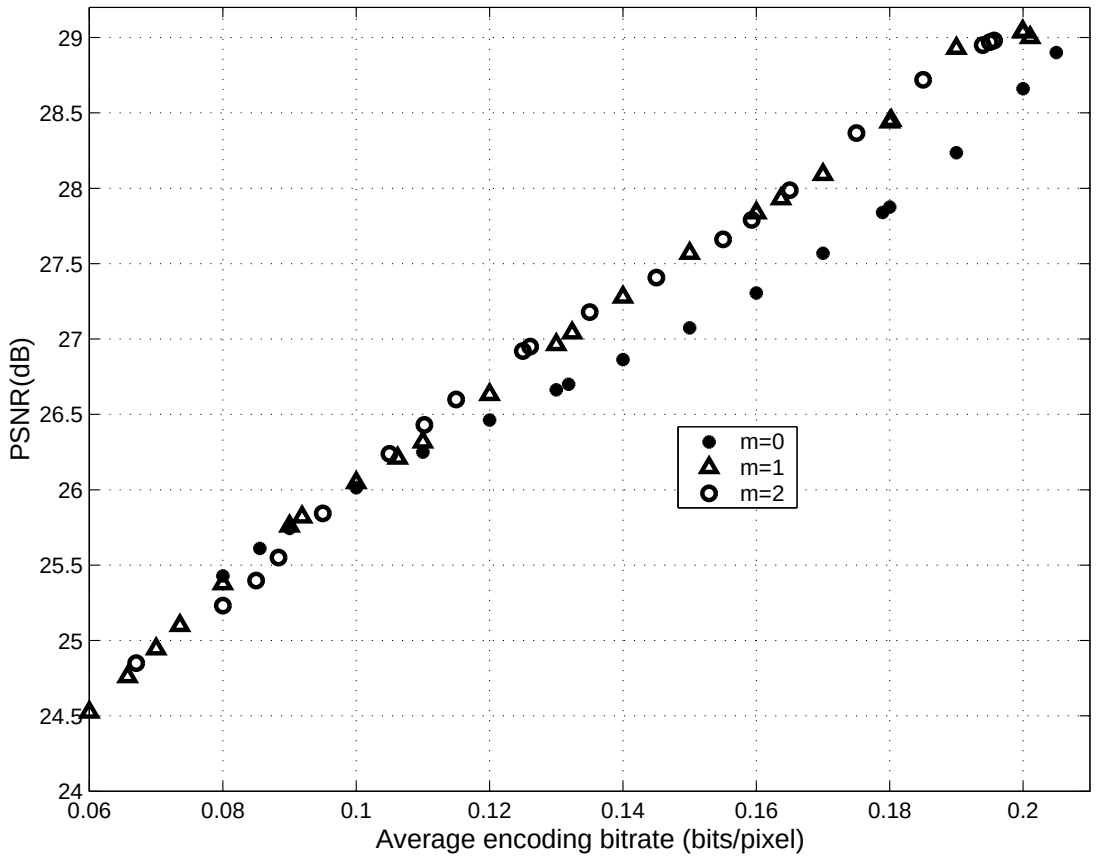
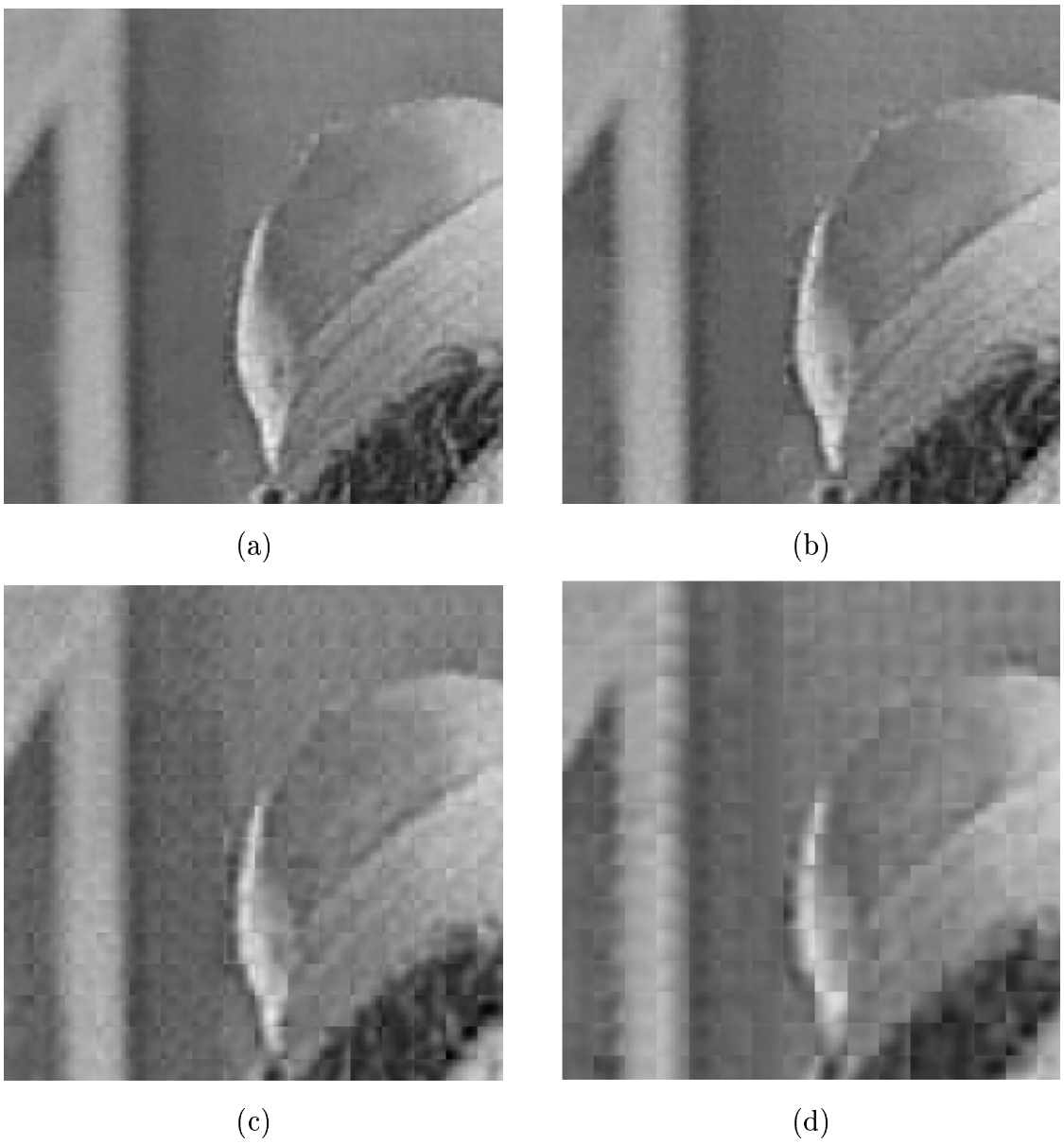


Figure 4.16: Rate-distortion performance of EC-RRVQ with 64 stages for the test image LENA at increasing values of m (The vector size is 16×16)



(a) (b)

(c) (d)

Figure 4.17: Cropped Image LENA coded using EC-RRVQ of dim=16 × 16 and $m = 1$ at a bit rate of (a) 0.201 bpp with PSNR of 29 dB (b) 0.164 bpp with PSNR of 27.83 dB (c) 0.106 bpp with PSNR of 26.11 dB (d) 0.066 bpp with PSNR of 24.76 dB

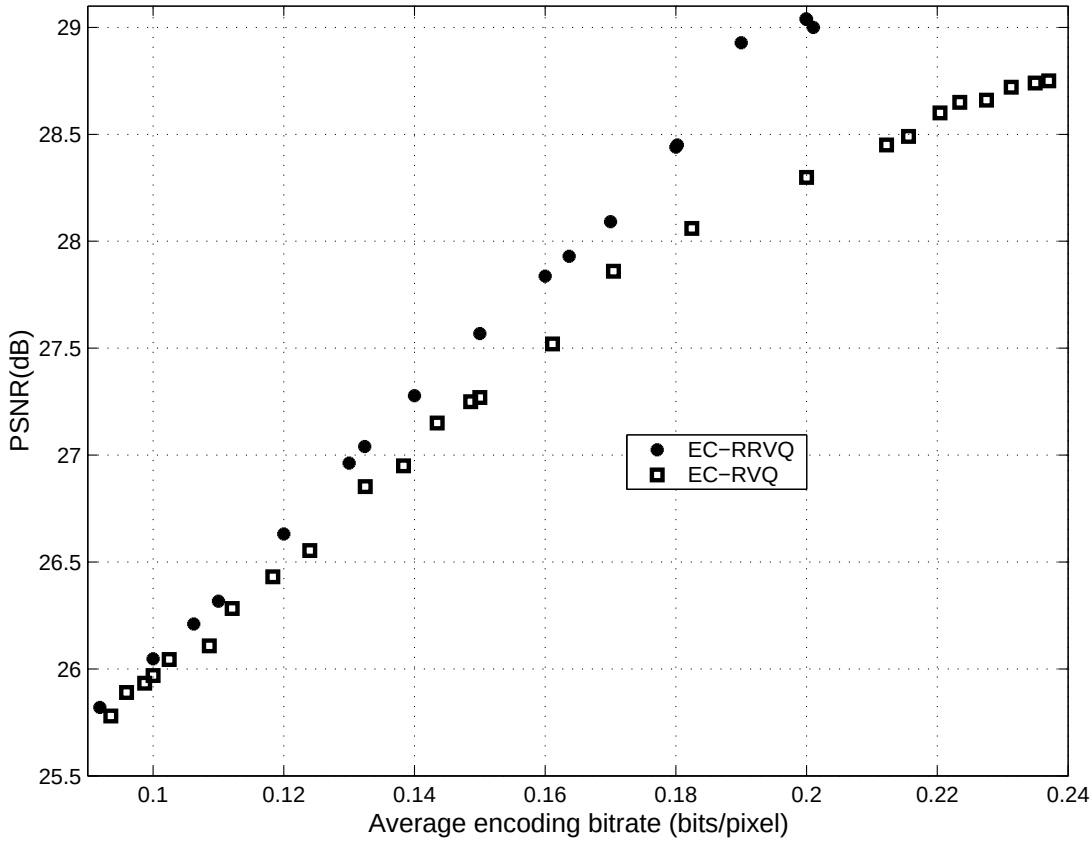
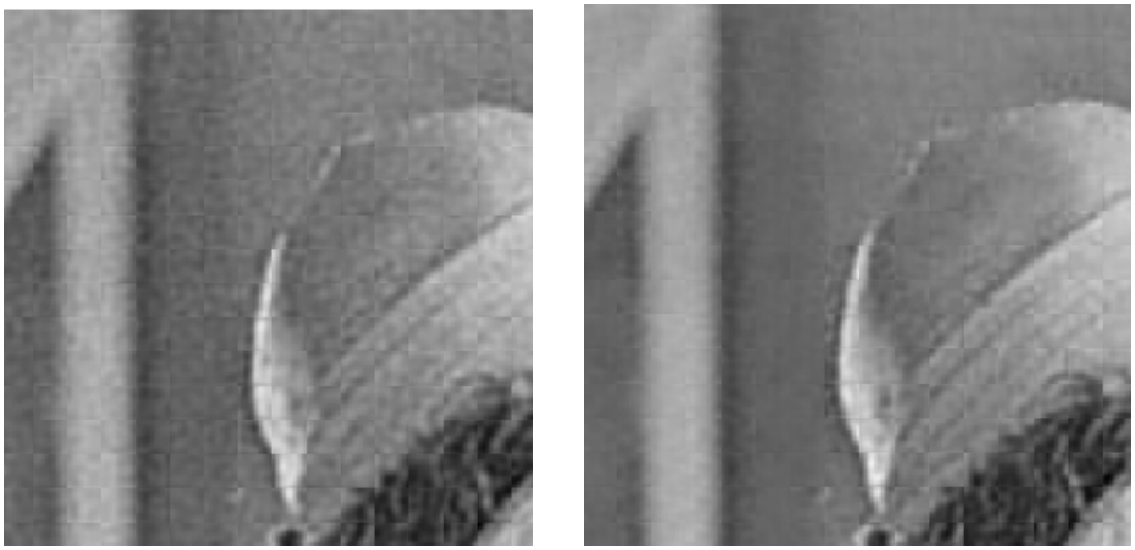


Figure 4.18: Rate-distortion performance of EC-RRVQ and EC-RVQ with 64 stages for the test image LENA at $m = 1$ (The vector size is 16×16)



(a) (b)

Figure 4.19: Cropped Image LENA coded using (a) EC-RVQ at a bit rate of 0.215 bpp with PSNR of 28.39 dB (b) EC-RRVQ at a bit rate of 0.201 bpp with PSNR of 29 dB both of dimension 16×16 and $m = 1$

Chapter 5

Conclusion

5.1 Summary of Thesis Contributions

In this thesis, an entropy-constrained reflected residual vector quantization (EC-RRVQ) design algorithm is introduced as an alternative to entropy-constrained residual vector quantization (EC-RVQ) and used to design codebooks for image coding.

EC-RRVQ is able to realize a more ordered or less random codebook, that offers two advantages. The first is that the RRVQ codebook, which is an ordered codebook, have a low output entropy while the second has to do with simplifying the search procedure used to find the best codeword. The idea discussed in this work is to introduce EC-RRVQ as a new baseline quantization scheme with single-path search and improved rate-distortion performance, which if joined with other trans-

form and subband coding methods will result in a competitive design. Because of single-path search, the EC-RRVQ has the potential to be a serious contender in the list of large block vector quantization implementation algorithms. Due to the RRVQ direct sum codevector structure which maintains a lower output entropy, EC-RRVQ can outperform EC-RVQ in rate-distortion performance, encoding complexity, and memory requirements. The EC-RRVQ has several advantages. Furthermore, the EC-RRVQ algorithm is stable as compared to the algorithm reported in [16]. The stability was verified based on empirical experiments. Also, the results obtained for high dimensional blocks such as the 8×8 and 16×16 vector sizes are competitive with the results reported in [43] and [42] in terms of rate-distortion performance, computational complexities and memory requirements.

The basic disadvantage of the EC-RRVQ is that it requires large training set vectors to achieve good rate-distortion performance. This is due to the fact that the RRVQ does not allow more than two codevectors per stage. Therefore, a large number of stages is needed for large bit rates.

5.2 Future Research Directions

An area of improvement can be effected by using adaptive entropy coding since the entropy tables are relatively small. Moreover, further improvement can be made by taking advantage of using the intervector dependencies and employing a high-order

entropy conditioning strategy that captures local information in the neighboring vectors. Finally, since in [44], a trellis-coded EC-RVQ (TC-ECRVQ) was shown to outperform EC-RVQ, it is expected that a trellis-coded EC-RRVQ will also outperform EC-RRVQ.

Appendix A

Mid-point derivation in the Lagrangian space

Consider

$$E[d(\boldsymbol{x}_p, \boldsymbol{y}_p(\boldsymbol{j}_p))] + \lambda L(j_p|j_{p-1}, j_{p-2}, \cdots, j_1) \tag{A.1}$$

to be the *pth*-stage Lagrangian where $\boldsymbol{x}_p$ is the residual vector, $\boldsymbol{y}_p(\boldsymbol{j}_p)$ is the p*th* stage codevector, and $L(j_p|j_{p-1}, j_{p-2}, \cdots, j_1)$ is the length of the codevector $\boldsymbol{y}_p(\boldsymbol{j}_p)$. Let the plane of equal Lagrangians at a given pth stage be defined as:

$$\begin{aligned} &\|\boldsymbol{x}_p - \boldsymbol{y}_p(1)\|^2 + \lambda L(1|j_{p-1}, j_{p-2}, \cdots, j_1) \\ = \;& \|\boldsymbol{x}_p - \boldsymbol{y}_p(2)\|^2 + \lambda L(2|j_{p-1}, j_{p-2}, \cdots, j_1) \end{aligned} \tag{A.2}$$

Then Eq. A.2 can be rewritten as:

$$\begin{aligned} &\|\boldsymbol{x}_p\|^2 - 2\boldsymbol{x}_p \cdot \boldsymbol{y}_p(1) + \|\boldsymbol{y}_p(1)\|^2 + \lambda L(1|j_{p-1}, j_{p-2}, \cdots, j_1) \\ &= \|\boldsymbol{x}_p\|^2 - 2\boldsymbol{x}_p \cdot \boldsymbol{y}_p(2) + \|\boldsymbol{y}_p(2)\|^2 + \lambda L(2|j_{p-1}, j_{p-2}, \cdots, j_1) \end{aligned} \tag{A.3}$$

Reordering Eq. A.3 and dividing by -2 both sides, one can obtain the following equation:

$$\begin{aligned} &(\boldsymbol{y}_p(1) - \boldsymbol{y}_p(2)) \cdot \boldsymbol{x}_p \\ &= \frac{\|\boldsymbol{y}_p(1)\|^2 - \|\boldsymbol{y}_p(2)\|^2}{2} \\ &+ \frac{\lambda L(1|j_{p-1}, j_{p-2}, \cdots, j_1)}{2} \\ &- \frac{\lambda L(2|j_{p-1}, j_{p-2}, \cdots, j_1)}{2} \end{aligned} \tag{A.4}$$

Therefore, using the normal plane equation $\boldsymbol{n}_p \cdot \boldsymbol{x}_p = d$, where $\boldsymbol{n}_p$ is a normal vector associated with the plane that join $\boldsymbol{y}_p(1)$ and $\boldsymbol{y}_p(2)$, i.e, $\boldsymbol{n}_p = \boldsymbol{y}_p(1) - \boldsymbol{y}_p(2)$, Eq. A.4 can be rewritten as:

$$\begin{aligned} d &= \frac{\|\boldsymbol{y}_p(1)\|^2 - \|\boldsymbol{y}_p(2)\|^2}{2} \\ &+ \frac{\lambda L(1|j_{p-1}, j_{p-2}, \cdots, j_1)}{2} \\ &- \frac{\lambda L(2|j_{p-1}, j_{p-2}, \cdots, j_1)}{2} \end{aligned} \tag{A.5}$$

Since $\boldsymbol{n}_p$ is perpendecular to the boundary between $\boldsymbol{y}_p(1)$ and $\boldsymbol{y}_p(2)$, then the shortest distance from $\boldsymbol{y}_p(2)$ to the plane of equal Lagrangian is given by:

$$B = \frac{|d| - \boldsymbol{n}_p \cdot \boldsymbol{y}_p(2)}{\|\boldsymbol{n}_p\|} \tag{A.6}$$

Now by substituting Eq. A.4 into A.6 and elaborating on the resulting equation, the boundary will be:

$$\begin{aligned} B = & \frac{\|\boldsymbol{y}_p(1) - \boldsymbol{y}_p(2)\|}{2} \\ + & \frac{\lambda L(1|j_{p-1}, j_{p-2}, \cdots, j_1)}{2\|\boldsymbol{y}_p(1) - \boldsymbol{y}_p(2)\|} \\ - & \frac{\lambda L(2|j_{p-1}, j_{p-2}, \cdots, j_1)}{2\|\boldsymbol{y}_p(1) - \boldsymbol{y}_p(2)\|} \end{aligned} \tag{A.7}$$

Hence, the mid-point for the EC-RRVQ case is:

$$\boldsymbol{m}_p = \boldsymbol{y}_p(2) + B\boldsymbol{n}_p \tag{A.8}$$

Therefore, the mid-point will be:

$$\begin{aligned} \boldsymbol{m}_p = & \boldsymbol{y}_p(2) + \frac{\|\boldsymbol{y}_p(1) - \boldsymbol{y}_p(2)\|}{2}\boldsymbol{n}_p \\ + & \frac{\lambda L(1|j_{p-1}, j_{p-2}, \cdots, j_1)}{2\|\boldsymbol{y}_p(1) - \boldsymbol{y}_p(2)\|}\boldsymbol{n}_p \\ - & \frac{\lambda L(2|j_{p-1}, j_{p-2}, \cdots, j_1)}{2\|\boldsymbol{y}_p(1) - \boldsymbol{y}_p(2)\|}\boldsymbol{n}_p \end{aligned} \tag{A.9}$$

Nomenclature

Abbreviations

VQ	Vector Quantization
RVQ	Residual Vector Quantization
RRVQ	Reflected Residual Vector Quantization
EC-RVQ	Entropy-constrained Residual Vector Quantization
EC-RRVQ	Entropy-constrained Reflected Residual Vector Quantization

English Symbols

P	Number of stages
N_p	The p^{th} stage codebook size
j_p	The p^{th} stage index: $\{1 \leq j_p \leq N_p\}$
$\boldsymbol{y}(j_p)$	The j_p^{th} codevector of the pth stage
$S(j_p)$	The j_p^{th} partition of the pth stage
$\mathcal{C}_p$	The p^{th} stage codebook
$\mathcal{P}_p$	The p^{th} stage partition
$\mathcal{Q}_p$	The p^{th} stage quantizer mapping
$\boldsymbol{m}_p$	The p^{th} stage mid-point
$\boldsymbol{n}_p$	The p^{th} stage normal vector
$\boldsymbol{z}_p$	A p^{th} stage point in the plane

Greek Symbols

λ	The lagrange multiplier

Bibliography

[1] T. Berger. *Rate Distortion Theory.* Prentice-Hall, Inc., New Jersey, 1971.

[2] R. G. Gallager. *Information Theory and Reliable Communications.* John Wiley & Sons, Inc, New York, 1968.

[3] R. M. Gray. *Vector quantization*, chapter 1, pages 4–29. 2. IEEE ASSP Magazine, April 1984.

[4] A. Gersho and R. Gray. *Vector quantization and signal compression.* Kluwer academic publishers, Boston, 1992.

[5] J. Makhoul, S. Roucos, and H. Gish. Vector quantization in speech coding. *Proc. IEEE*, 73:1551–1588, 1985.

[6] C. E. Shannon. A mathematical theory of communication. *Bell Sys. Tech. Journal*, 27:379–423, 623–656, 1948.

[7] C. F. Barnes and R. L. Frost. Vector quantizers with direct sum codebooks. *IEEE transactions on information theory*, 39(2):565–580, March 1993.

[8] C.F. Barnes, S.A. Rizvi, and N.M. Nasrabadi. Advances in residual vector quantization: a review. *IEEE transactions on image processing*, 5(2):226–262, February 1996.

[9] B. Juang and A. Gray. Multiple stage vector quantization for speech coding. In *Proceedings of ICASSP-82*, volume 1, pages 597–600, Apr 1982.

[10] C. F. Barnes. *Residual Quantizers.* PhD thesis, Brigham Young University, Provo, Utah, 1989.

[11] T. Lookabaugh, E. A. Riskin, P. A. Chou, and R. M. Gray. Variable rate vector quantization for speech, image and video compression. *IEEE trans. on comm.*, 41:186–199, Jan. 1993.

[12] F. Kossentini, M.J.T. Smith, and C.F. Barnes. Necessary conditions for the optimality of variable-rate residual vector quantizers. *IEEE trans. on information theory*, 41(6):1903–1914, November 1995.

[13] P.A. Chou, T. Lookabaugh, and R.M. Gray. Entropy-constrianed vector quantization. *IEEE tarn. on acoustic,speech, and signal processing*, 37(1):31–42, Jan. 1989.

[14] J. Pan and T.R. Fischer. Two-stage vector qunatization-lattice vector quantization. *IEEE trans. on info. theory*, 41(1):155–163, Jan. 1995.

[15] W. A. H. Mousa and M. A. U. Khan. Design and analysis of entropy-constrained reflected residual vector quantization. *Proceedings of IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, III:2529–2532, May 2002.

[16] N. Farvardin and J. W. Modestino. Optimum quantizer performance for a class of non-gaussian memoryless sources. *IEEE Tran. on Info. Theory*, 30(3), May 1984.

[17] M. J. T. Smith and A. Docef. *A Study Guide for Digital Image Processing.* Scientific Publishers, Inc, 1999.

[18] T. Cover and J. Thomas. *Elements of Information Theory.* John Wiley & Sons, 1991.

[19] C. E. Shannon. Coding theorems for a discrete source with a fidelity criterion. *In IRE National Convention Record*, 4:142–163, 1959.

[20] Robert M. Gray and David L. Neuhoff. Quantization. *IEEE Tran. on Info. Theory*, 44(6), 1998.

[21] T. Lookabaugh, E. A. Riskin, P.A. Chou, and R.M. Gray. Variable rate vector quantization for speech, image and video compression. *IEEE transactions on communications*, 41(1):186–199, Jan 1993.

[22] A. Buzo, A. H. Gray Jr., R. M. Gray, and J. D. Markel. Speech coding based upon vector quantization. *IEEE Trans. on Acous. Speech and Signal Processing*, ASSP-28(5):562–574, 1980.

[23] M. J. Sabin and R. M. Gray. Product code vector quantizers for waveform and voice coding. *IEEE Trans. Acoust. Speech. Signal Process.*, (ASSP-32):474–488, June 1984.

[24] F. Kossentini, M.J.T. Smith, and C.F. Barnes. Image coding using entropy-constarined residual vector quantization. *IEEE transactions on image processing*, 4(10):1349–1356, October 1995.

[25] W. R. Bennett. Spectra of quantized signals. *Bell Systems Technical Journal*, pages 27:446–472, July 1948.

[26] K. Sayood. *Introduction to Data Compression*. Morgan Kaufmann Publishers, 2nd edition, 2000.

[27] Y. Linde, A. Buzo, and R. M. Gray. An algorithm for vector quantizer design. *IEEE Transactions on Communications*, pages 84–95, January 1980.

[28] T. Kohonen. The self-organizing map. *Proceedings of IEEE*, 78(9):1464–1480, 1990.

[29] M. E. Petersen, D. de Ridder, and H. Handels. Image processing with neural networks-a review. *The Journal of The Pattern Recognition Society, Pergamon*, 2002.

[30] J. McAuliffe, L. Atlas, and C. Rivera. A comparison of the lbg algorithm and kohonen neural network paradigm for image vector quantization. *Proc. of Int. Conf. on Acoustics, Speech and Signal Processing*, 4:2293–2296, 1990.

[31] R. M. Gray. *Source Coding Theorem*. Kluwer Academic Publishers, 1990.

[32] E. A. Riskin. *Variable Rate Vector Quantization for Images*. PhD thesis, Stanford University, 1990.

[33] J.H. Conway and N.J.A. Sloane. Voroni regions of lattices, second moments of polytopes, and quantization. *IEEE Tran. on Info. Theory*, 28:211–226, March 1982.

[34] J. H. Conway and N. Sloane. Fast quantizing and decoding algorithms for lattice quantizers and codes. *IEEE Transactions on Information theory*, IT-28:227–232, March 1982.

[35] S. E. Budge. Vector quantization of color digital images using product codes. Master's thesis, Brigham Young University, Provo, Utah, 1985.

[36] R. L. Baker. *Vector quantization of digital images*. PhD thesis, Standford University, 1984.

[37] E. A. Riskin, T. Lookabaugh, P. A. Chou, and R. M. Gray. Variable rate vector quantization for medical image compression. *IEEE Tran. on Medical Imaging*, 9:290–298, 1990.

[38] T. Saito and H. Takeo. Gain/shape vector quantization for multidimensional spherically symmetric random source. *Electornics and Communications in Japan*, 1986.

[39] Saito and et al. Adaptive discrete cosine transform image coding using gain/shape vector quantizers. In *Proc. IEEE Int. Conf. on Communications*, pages 1285–1289, 1986.

[40] R. C. Gonzalez and R. E. Woods. *Digital Image Processing.* Addison-Wesley, 1993.

[41] M. A. U. Khan and W. A. H. Mousa. Image coding using entropy-constrained reflected residual vector quantization. *Proceedings of IEEE Int. Conf. on Image Processing (ICIP)*, I:253–256, Sept. 2002.

[42] F. Kossentini, M. J. T. Smith, and C. F. Barnes. Large block rvq with multipath searching. In *Proceedings of IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, 1992.

[43] F. Kossentini, W.C. Chung, and M.J.T. Smith. Conditional entropy-constrained residual vq with application to image coding. *IEEE transactions on image processing*, 5(2):311–320, Feb 1996.

[44] M.A. Khan, M.J.T. Smith, and S.W. Mclaughlin. Trellis-coded residual vector quantization with application to image coding. In *proceedings of IEEE International symposium on circuits and systems*, Orlando, Florida, Jun 1999.

Vita

Wail Abdul-Hakim Mousa was born in Makkah, Saudi Arabia on September 20, 1977. In September 2001, he married Roba O. Rakkah. He received Bachelor of Science (B.S) degree in Electrical Engineering in summer 2000 with honors. In addition, he received a B.S in Mathematics (double major) in Fall 2002 also with honors. Both degrees were obtained from King Fahd University of Petroleum and Minerals (KFUPM). After he finished his Electrical Engineering (EE) B.S, he joined the EE Department at KFUPM as a Graduate Assistant in Fall 2001. Later on, in Fall 2002, he started his Master of Science (M.S) at the EE Department at KFUPM. He finished his M.S degree in Spring 2003.

His research interests include image compression and classification, mathematical image processing, and signal & image processing. He is a student member of the IEEE and a member of the IEEE Signal Processing society. He enjoys swimming and playing football.